Computer Vision WS 14/15

Computer Vision – Lecture 22

Repetition

03.02.2015

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Announcements

- Exam
  - 1st Date: Monday, 23.02., 13:30 - 17:30h
  - 2nd Date: Thursday, 26.03., 09:30 - 12:30h
  - Closed-book exam, the core exam time will be 2h.
  - Admission requirement: 50% of the exercise points or passed test exam
  - We will send around an announcement with the exact starting times and places by email.
- Test exam
  - Date: Thursday, 05.02., 09:15 - 10:45h, room UMIC 025
  - Core exam time will be 1h
  - Purpose: Prepare you for the questions you can expect.
  - *Possibility to collect bonus exercise points!*

Announcements (2)

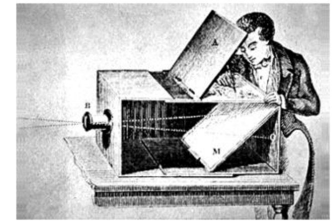
- **Feedback to the lecture evaluation**

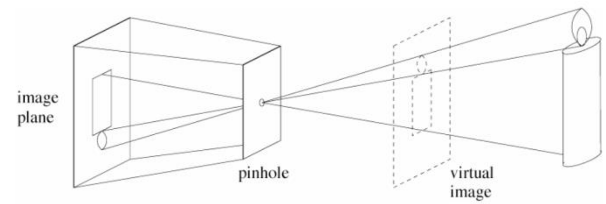
Announcements (3)

- **Today, I'll summarize the most important points from the lecture.**
  - It is an opportunity for you to ask questions…
  - …or get additional explanations about certain topics.
  - *So, please do ask*.
- **Today's slides are intended as an index for the lecture.**
  - But they are not complete, won't be sufficient as only tool.
  - Also look at the exercises - they often explain algorithms in detail.

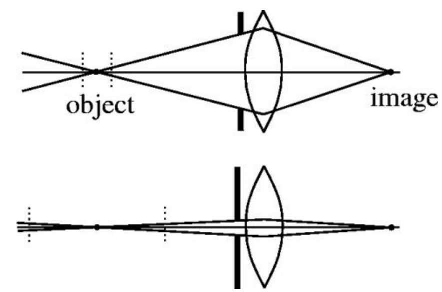
Repetition

- Image Processing Basics
  - Image Formation
  - Binary Image Processing
  - Linear Filters
  - Edge & Structure Extraction
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

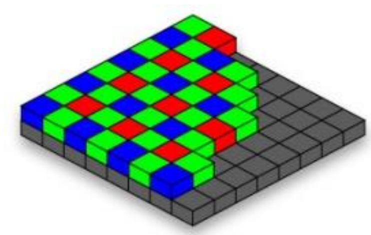




Pinhole camera model

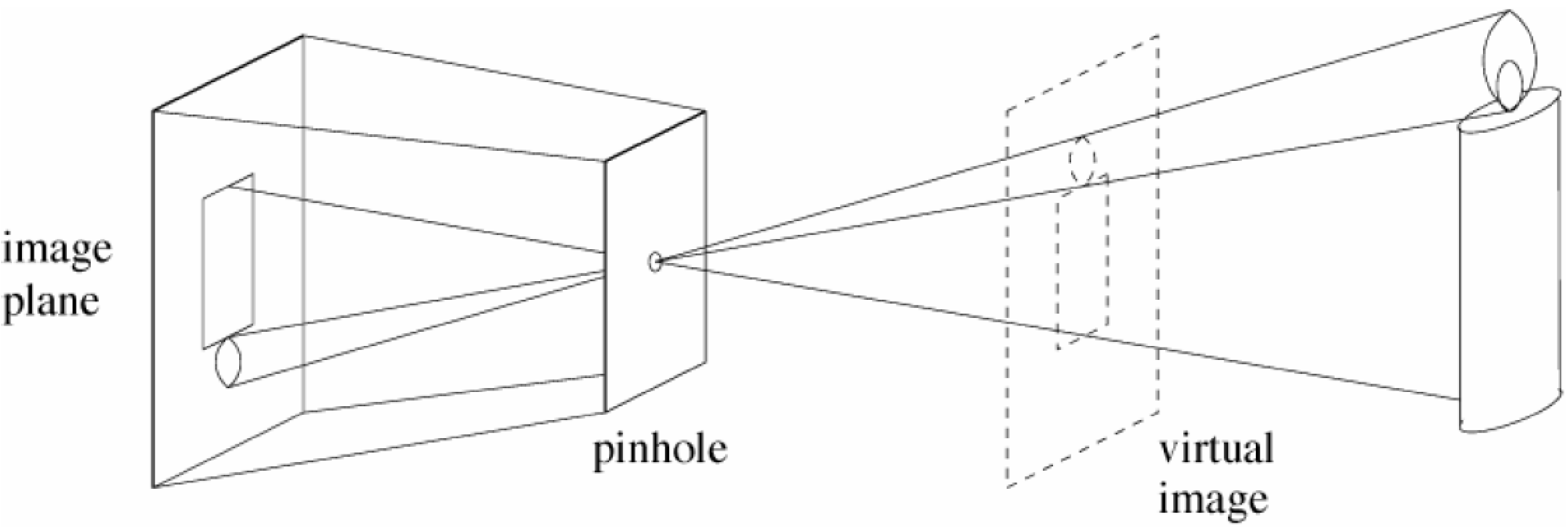


Lenses, focal length, aperture


Color sensors

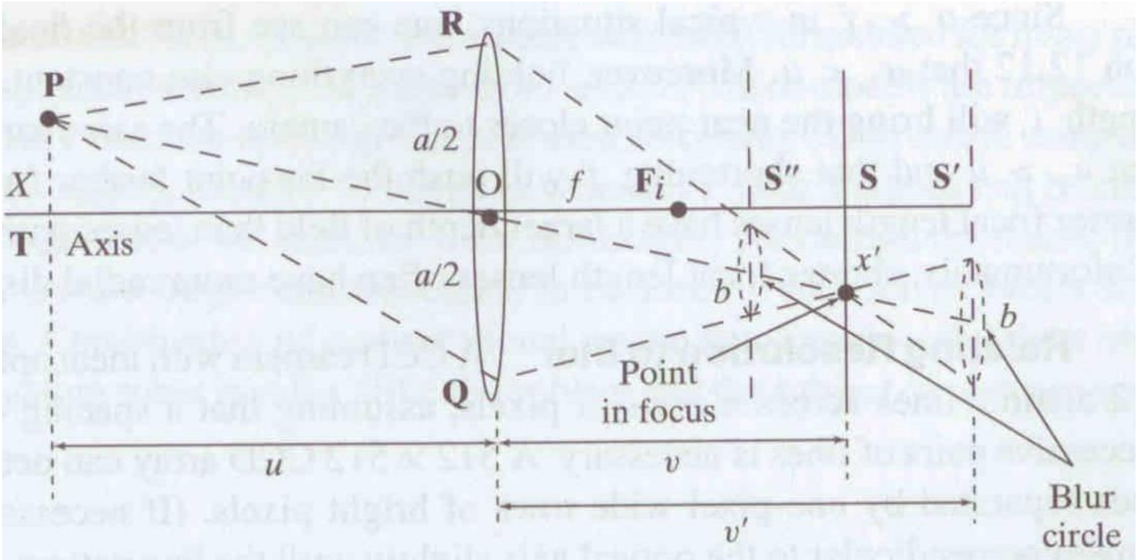
Recap: Pinhole Camera

- (Simple) standard and abstract model today
  - Box with a small hole in it
  - Works in practice



Source: Forsyth & Ponce

Recap: Focus and Depth of Field



"circles of confusion"

Thin lens: scene points at distinct depths come in focus at different image planes.

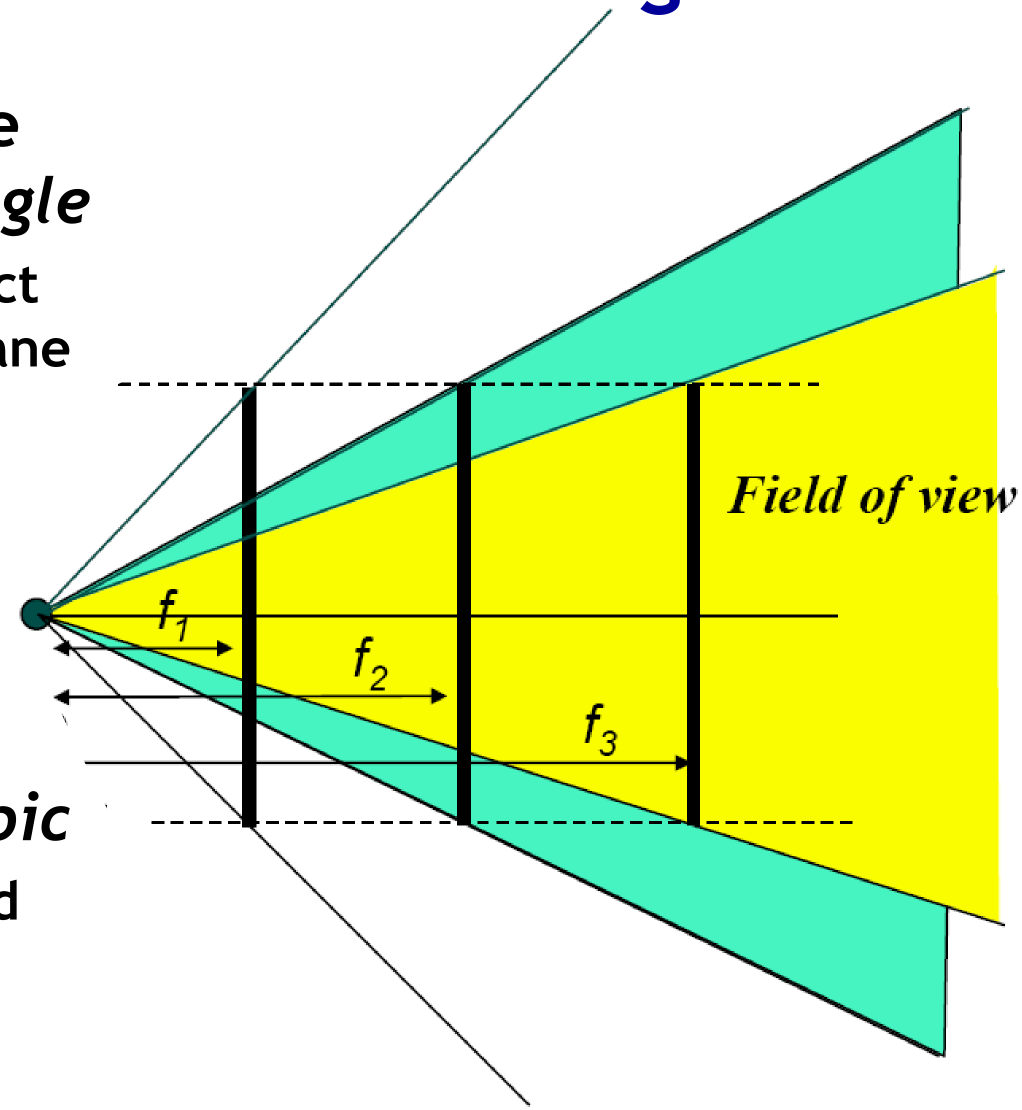
(Real camera lens systems have greater depth of field.)

- Depth of field: distance between image planes where blur is tolerable

Source: Shapiro & Stockman

Recap: Field of View and Focal Length

- As *f* gets smaller, image becomes more *wide angle*
  - More world points project onto the finite image plane

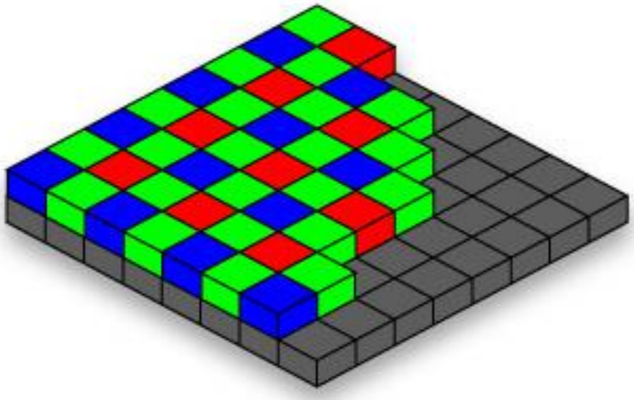


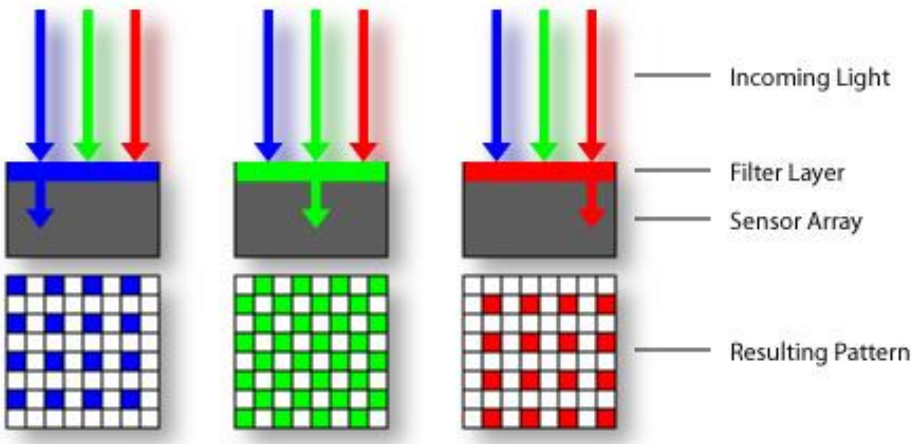
- As *f* gets larger, image becomes more *telescopic*
  - Smaller part of the world projects onto the finite image plane

from R. Duraiswami

Recap: Color Sensing in Digital Cameras

Bayer grid



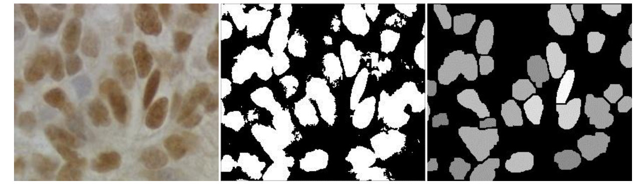


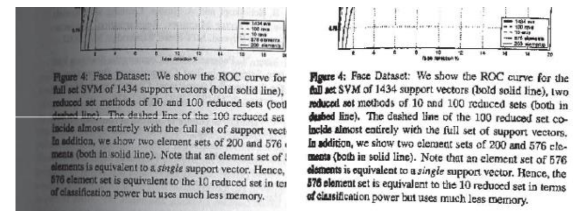
Estimate missing components from neighboring values (demosaicing)

Source: Steve Seitz

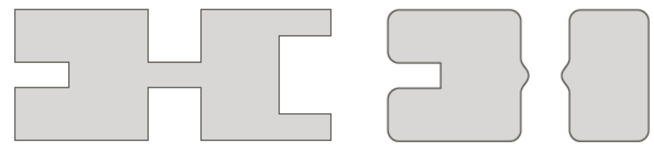
Repetition

- Image Processing Basics
  - Image Formation
  - Binary Image Processing
  - Linear Filters
  - Edge & Structure Extraction
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking



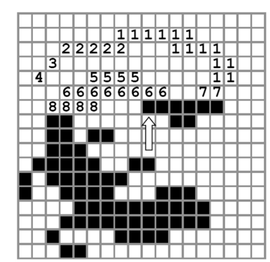


Thresholding



$A \circ B = (A \ominus B) \oplus B$

Morphological Operators



Connected Components

Recap: Binary Processing Pipeline

see "Haribo" Demo!

- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract individual objects
  - Connected Components Labeling
- Describe the objects
  - Region properties

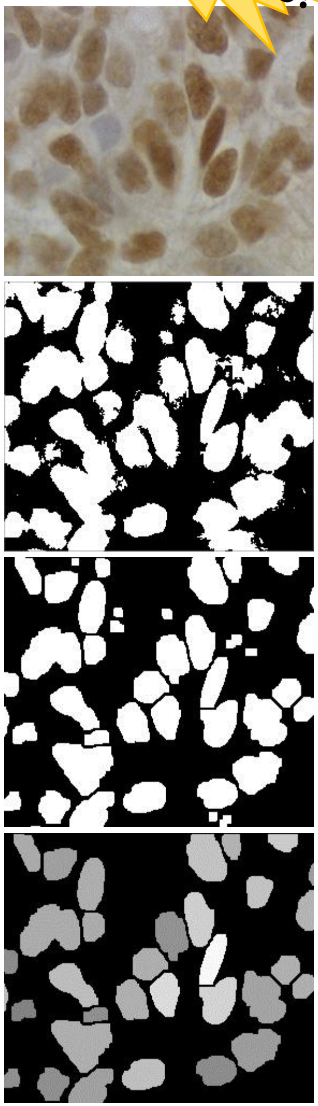
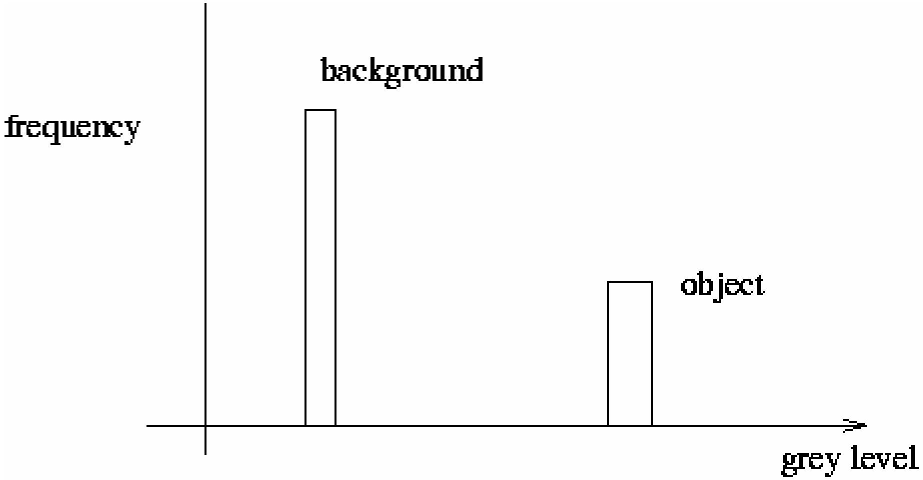
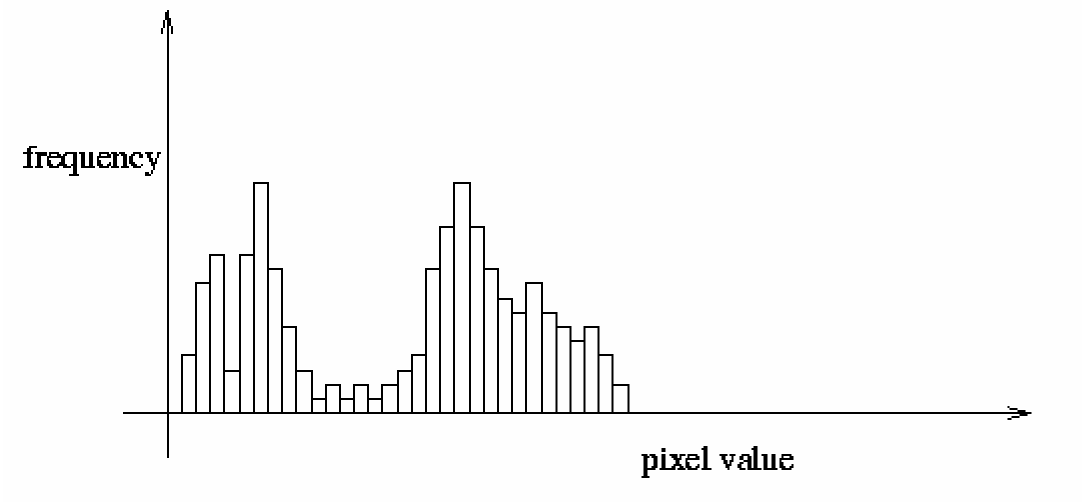


Image Source: D. Kim et al., Cytometry 35(1), 1999

Recap: Robust Thresholding



Ideal histogram, light object on dark background



Actual observed histogram with noise

Assumption here: only two modes

Source: Robyn Owens

Recap: Global Binarization [Otsu'79]

see Exercise 2.1!

- **Precompute a cumulative grayvalue histogram h.**
- **For each potential threshold T**
  - 1.) Separate the pixels into two clusters according to T.
  - 2.) Compute both cluster means $\mu_1(T)$ and $\mu_2(T)$.
    Look up n_1, n_2 in h

$$n_1(T) = |\{I_{(x,y)} < T\}|, \quad n_2(T) = |\{I_{(x,y)} \geq T\}|$$

  - 3.) Compute the between-class variance $\sigma^2_{between}(T)$

$$\sigma^2_{between}(T) \;=\; n_1(T)n_2(T)\left[\mu_1(T) - \mu_2(T)\right]^2$$

- **Choose the threshold that maximizes**

$$T^* = \arg\max_T \left[\sigma^2_{between}(T)\right]$$

Recap: Background Surface Fitting

- Document images often contain a smooth gradient

⇒ *Try to fit that gradient with a polynomial function*

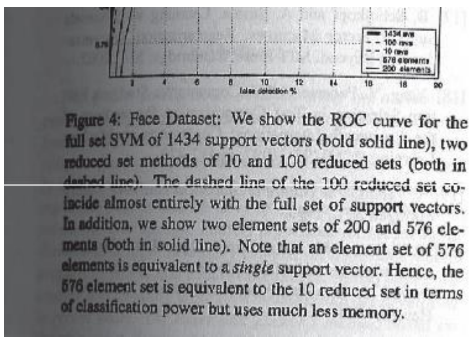
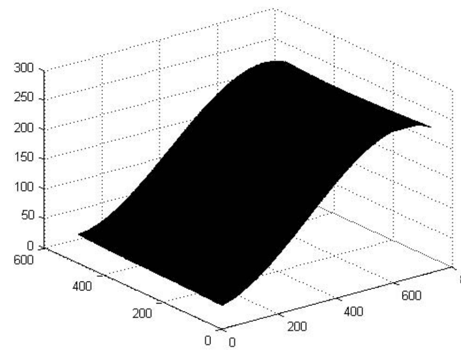
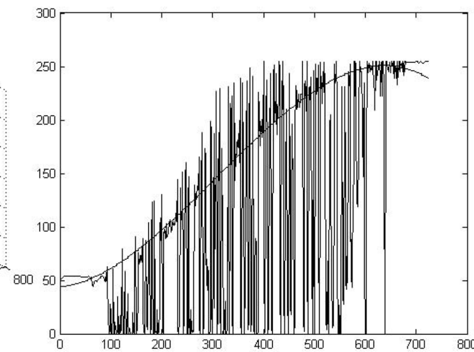


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

Original image





Fitted surface

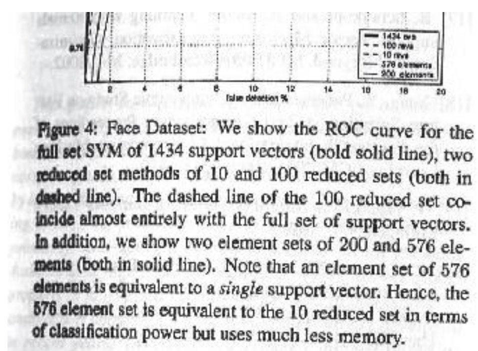


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

Shading compensation

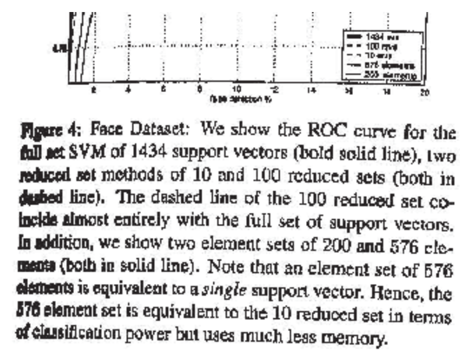


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

Binarized result

Source: S. Lu & C. Tan, ICDAR'07

Recap: Dilation

- **Definition**
  - *"The dilation of A by B is the set of all displacements z, such that $(\hat{B})_z$ and A overlap by at least one element".*
  - ($(\hat{B})_z$ is the mirrored version of B, shifted by z)
- **Effects**
  - If current pixel z is foreground, set all pixels under $(B)_z$ to foreground.

⇒ Expand connected components

⇒ Grow features

⇒ Fill holes

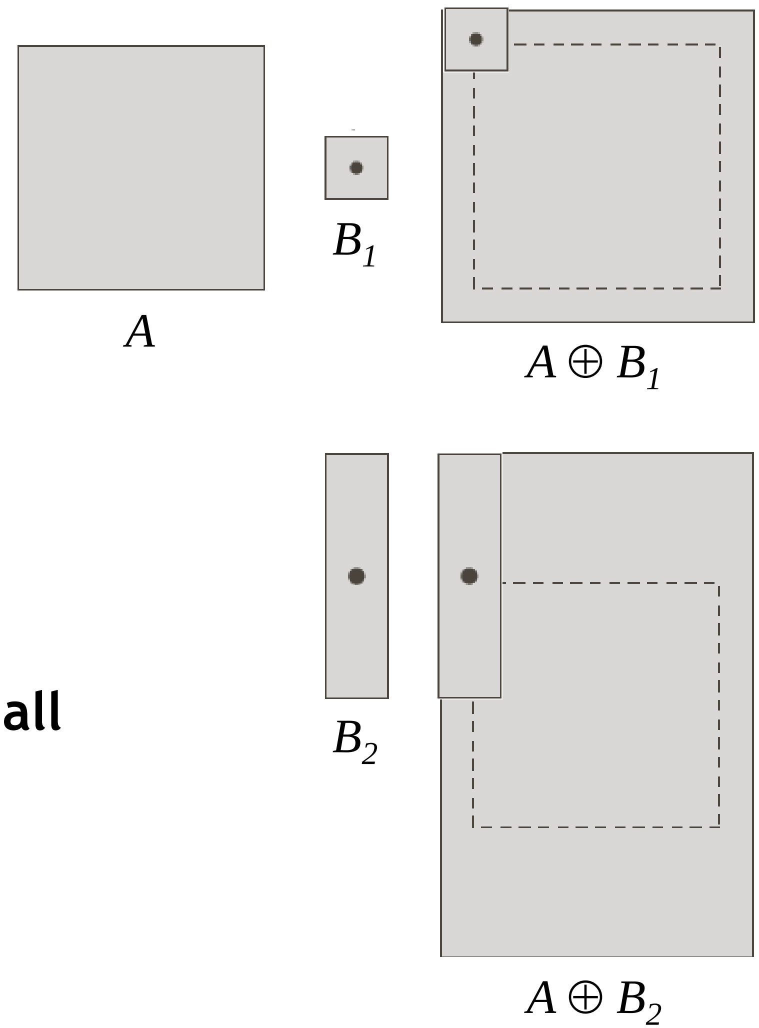


Image Source: R.C. Gonzales & R.E. Woods

Recap: Erosion

- **Definition**
  - *"The erosion of A by B is the set of all displacements z, such that $(B)_z$ is entirely contained in A".*

- **Effects**
  - **If not every pixel under $(B)_z$ is foreground, set the current pixel z to background.**

  $\Rightarrow$ **Erode connected components**

  $\Rightarrow$ **Shrink features**

  $\Rightarrow$ **Remove bridges, branches, noise**

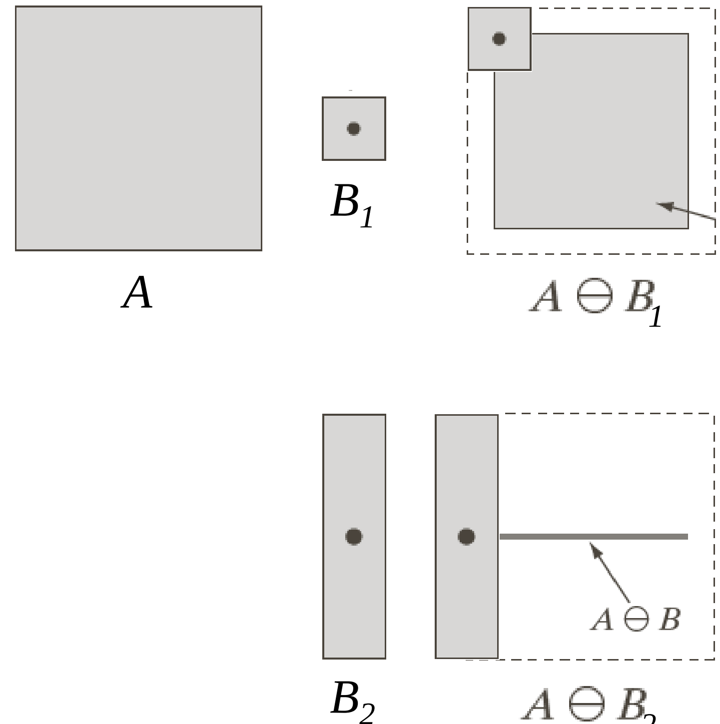


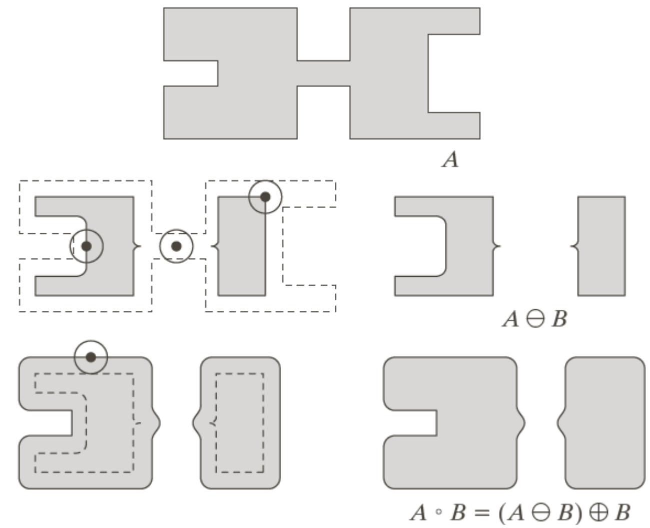
Image Source: R.C. Gonzales & R.E. Woods

RWTH AACHEN UNIVERSITY

Recap: Opening

- **Definition**
  - Sequence of **Erosion and Dilation**

    $A \circ B = (A \ominus B) \oplus B$



- **Effect**
  - $A \circ B$ is defined by the points that are reached if B is ***rolled around inside*** A.

  $\Rightarrow$ Remove small objects, keep original shape.

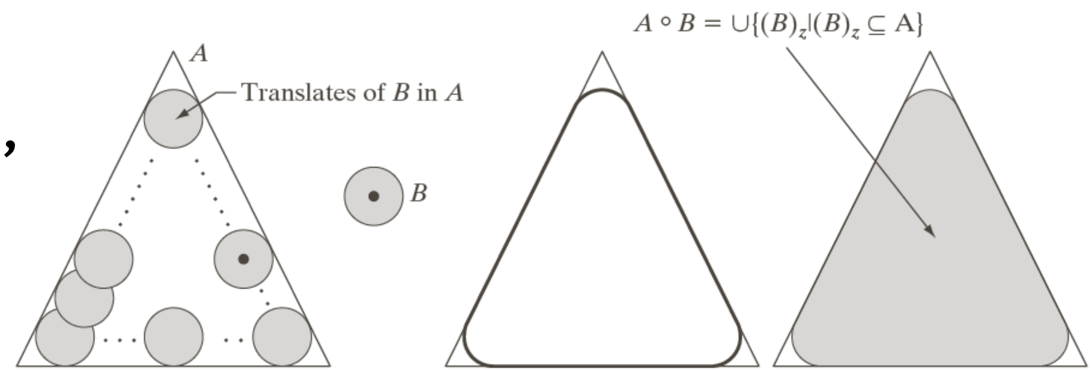
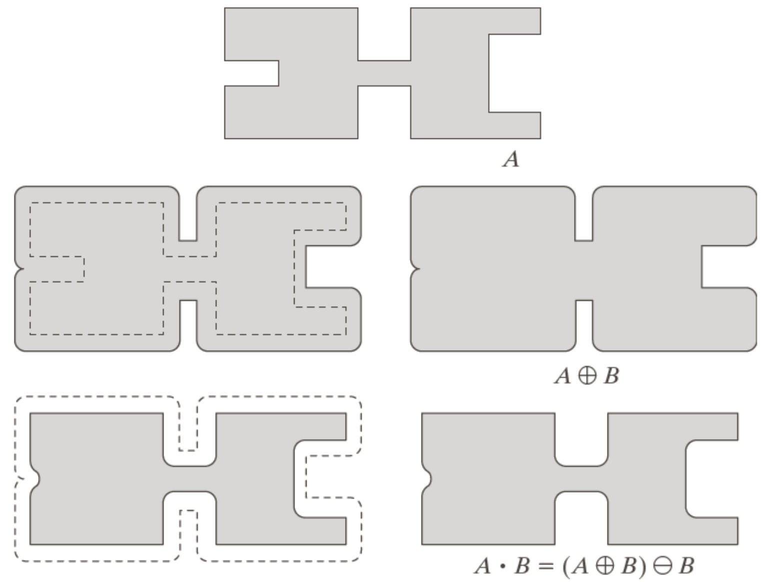


Image Source: R.C. Gonzales & R.E. Woods

Recap: Closing

- **Definition**
  - Sequence of **Dilation and Erosion**

    $A \bullet B = (A \oplus B) \ominus B$



- **Effect**
  - $A \bullet B$ is defined by the points that are reached if B is *rolled around on the outside* of A.

  $\Rightarrow$ Fill holes,
  keep original shape.

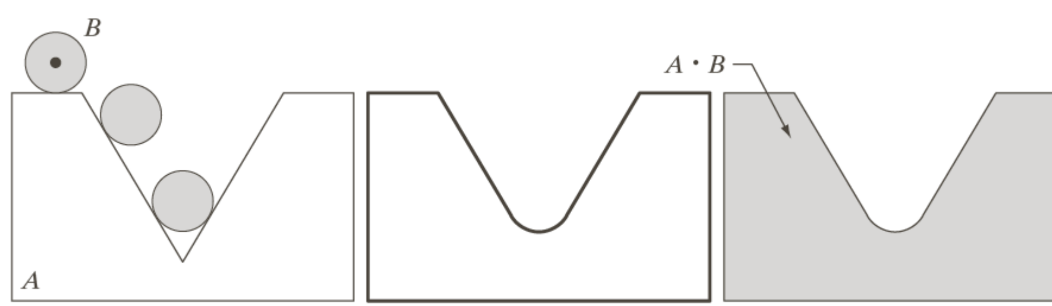
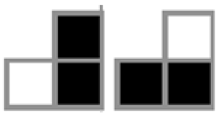


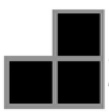
Image Source: R.C. Gonzales & R.E. Woods

Recap: Connected Components Labeling

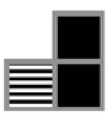
- **Process the image from left to right, top to bottom:**
  - 1.) If the next pixel to process is 1



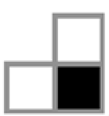
i.) If only one of its neighbors (top or left) is 1, copy its label.



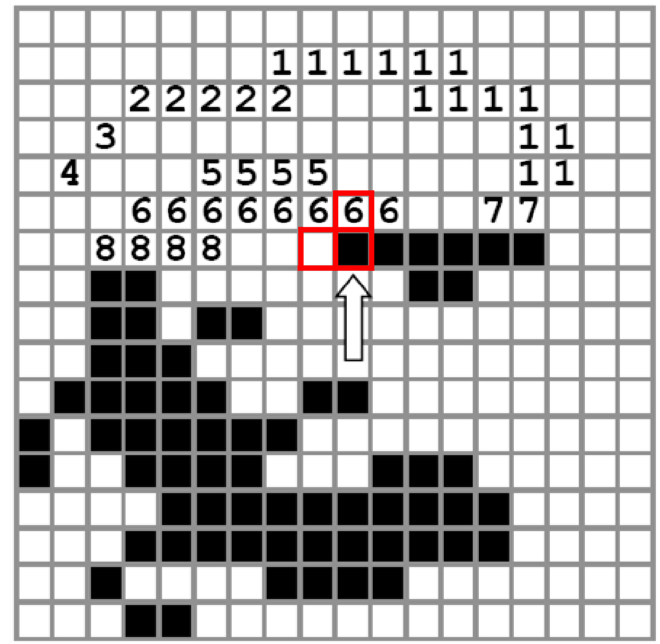
ii.) If both are 1 and have the same label, copy it.



iii.) If they have different labels
  - – Copy the label from the left.
  - – Update the equivalence table.



iv.) Otherwise, assign a new label.



- **Re-label with the smallest of equivalent labels**

```
{1, 2, 7}
{3}
{4}
{5, 6, 8}
{}
```

Recap: Region Properties

- From the previous steps, we can obtain separated objects.
- Some useful features can be extracted once we have connected components, including
  - Area
  - Centroid
  - Extremal points, bounding box
  - Circularity
  - Spatial moments

Recap: Moment Invariants

see Exercise 2.3!

- Normalized central moments

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \qquad \gamma = \frac{p+q}{2} + 1$$

- From those, a set of *invariant moments* can be defined for object description.

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

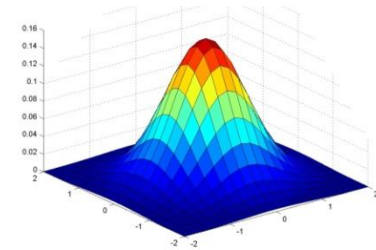
$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

(Additional invariant moments ϕ_5, ϕ_6, ϕ_7 can be found in the literature).

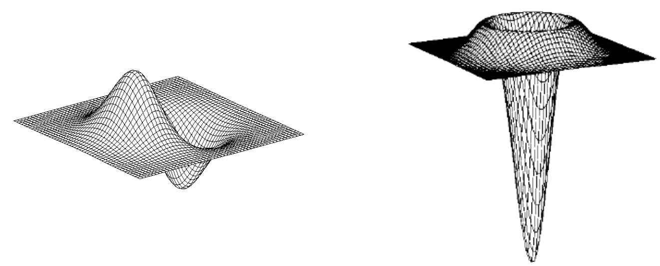
- Robust to translation, rotation & scaling, but don't expect wonders (still summary statistics).

Repetition

- Image Processing Basics
  - Image Formation
  - Binary Image Processing
  - Linear Filters
  - Edge & Structure Extraction
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking



Gaussian Smoothing

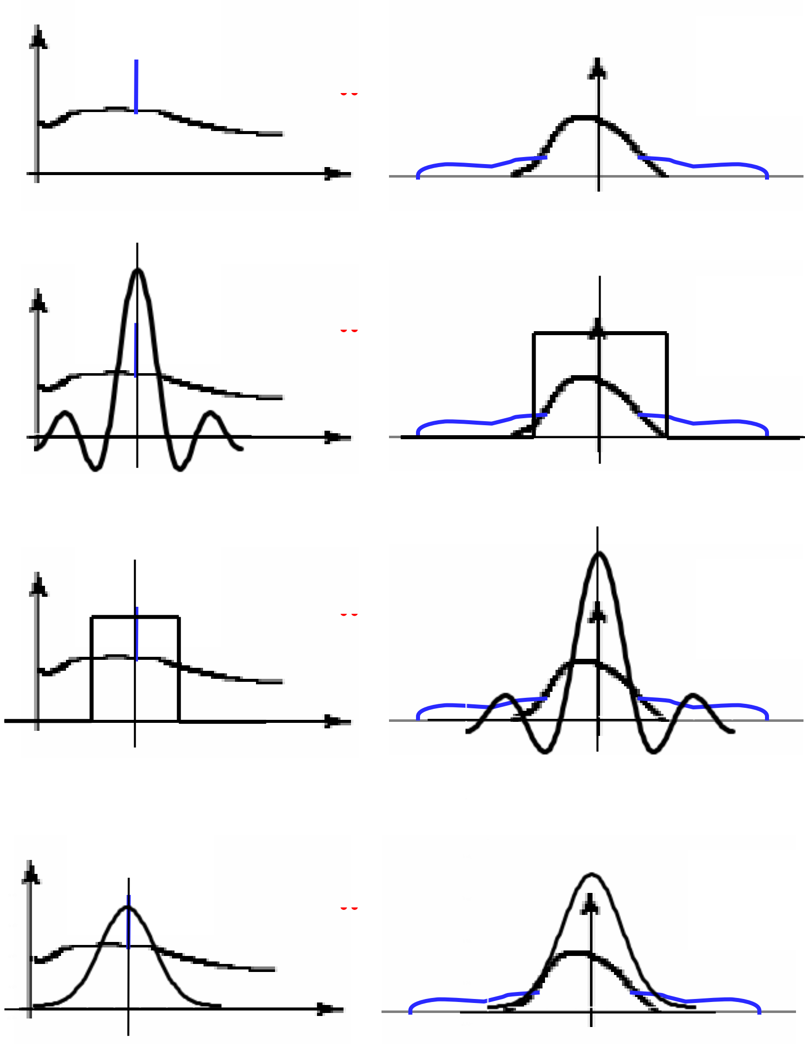


Derivative operators

Gaussian/Laplacian pyramid

Recap: Effect of Filtering

- Noise introduces high frequencies. To remove them, we want to apply a "low-pass" filter.
- The ideal filter shape in the frequency domain would be a box. But this transfers to a spatial sinc, which has infinite spatial support.
- A compact spatial box filter transfers to a frequency sinc, which creates artifacts.
- A Gaussian has compact support in both domains. This makes it a convenient choice for a low-pass filter.



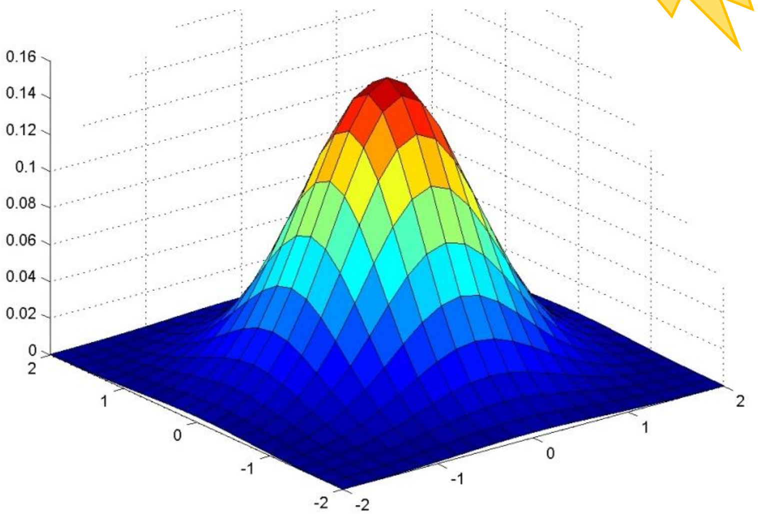
RWTH AACHEN UNIVERSITY

Recap: Gaussian Smoothing

see Exercise 2.4!

- Gaussian kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$



- Rotationally symmetric
- Weights nearby pixels more than distant ones
  - This makes sense as 'probabilistic' inference about the signal
- A Gaussian gives a good model of a fuzzy blob

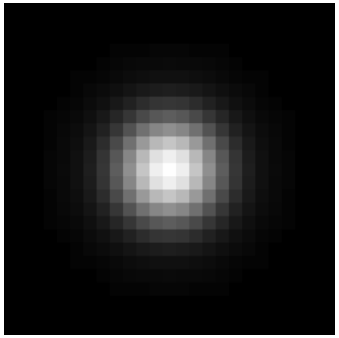
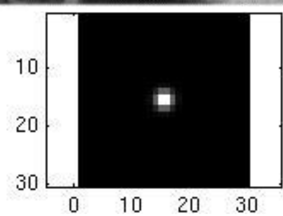
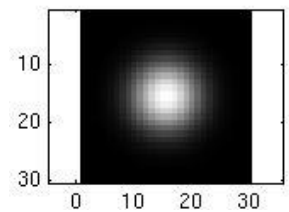


Image Source: Forsyth & Ponce

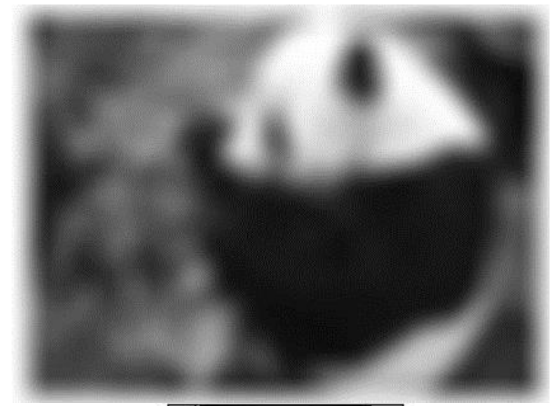
Recap: Smoothing with a Gaussian

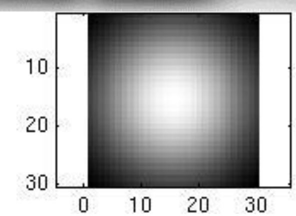
- Parameter σ is the "scale" / "width" / "spread" of the Gaussian kernel and controls the amount of smoothing.





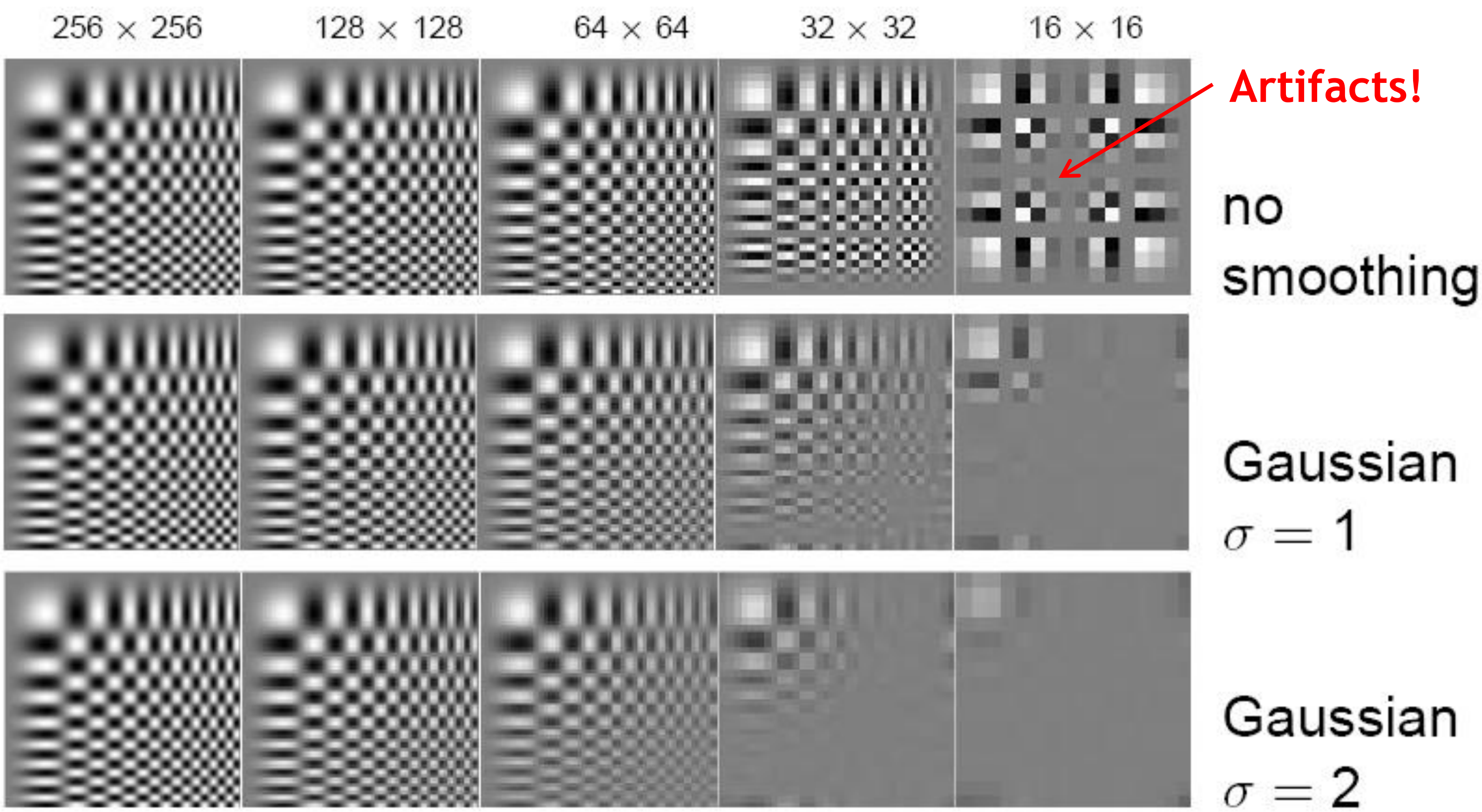
…






```
for sigma=1:3:10
   h = fspecial('gaussian`, fsize, sigma);
   out = imfilter(im, h);
   imshow(out);
   pause;
end
```

Recap: Resampling with Prior Smoothing

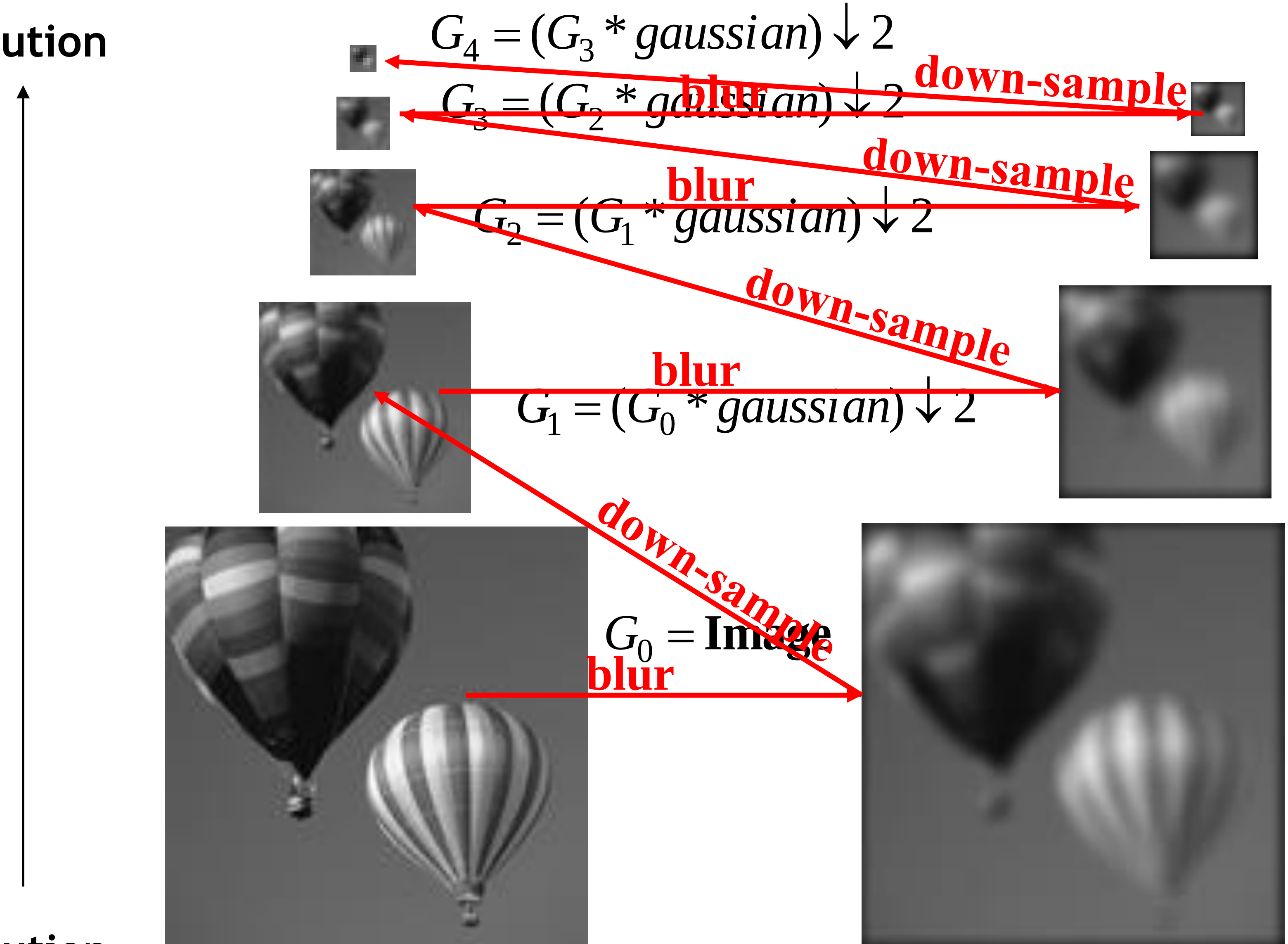


- **Note: We cannot recover the high frequencies, but we can avoid artifacts by smoothing before resampling.**

Image Source: Forsyth & Ponce

Recap: The Gaussian Pyramid

Low resolution

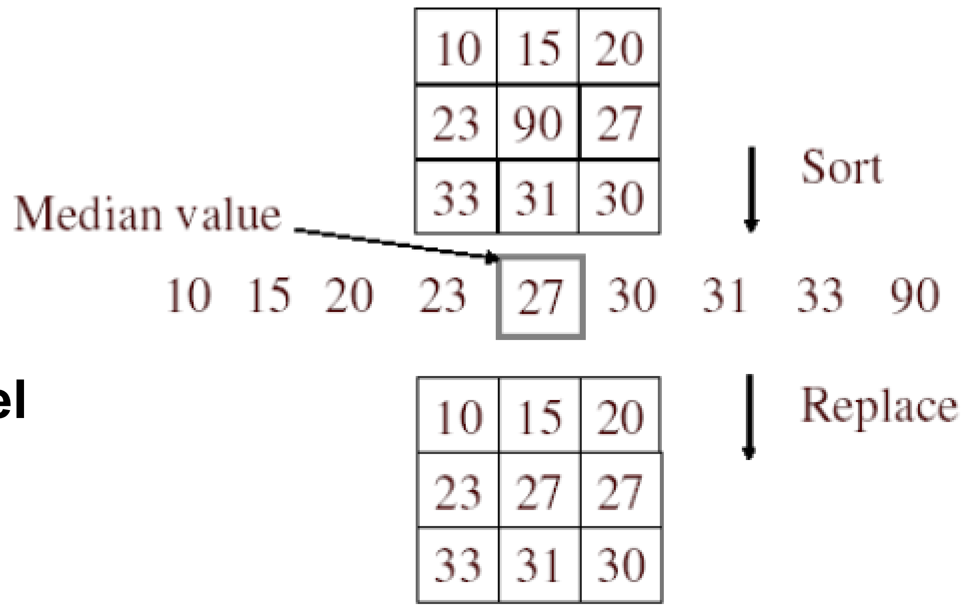


High resolution

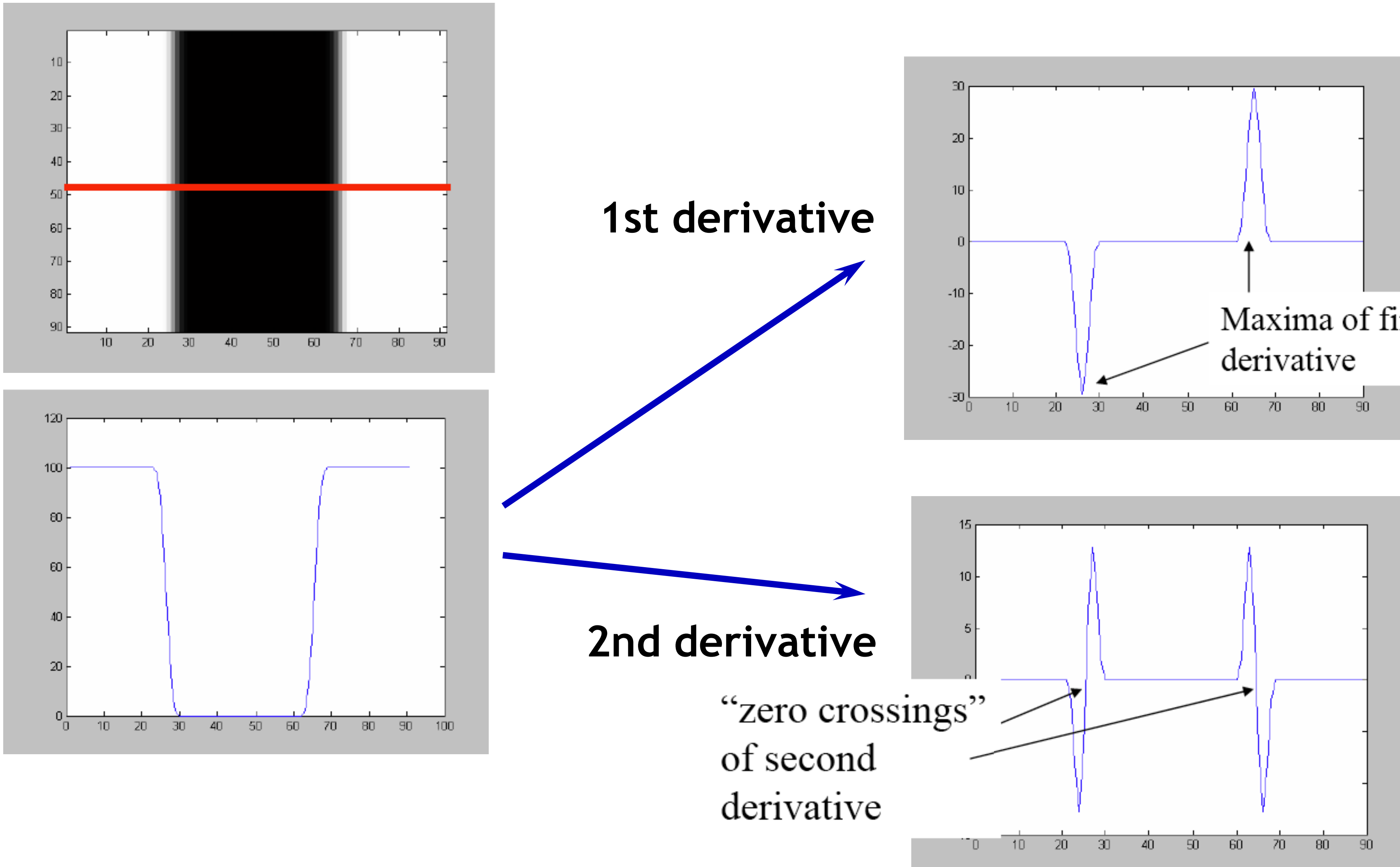
Source: Irani & Basri

Recap: Median Filter

- Basic idea
  - Replace each pixel by the median of its neighbors.
- Properties
  - Doesn't introduce new pixel values
  - Removes spikes: good for impulse, salt & pepper noise
  - Nonlinear
  - Edge preserving

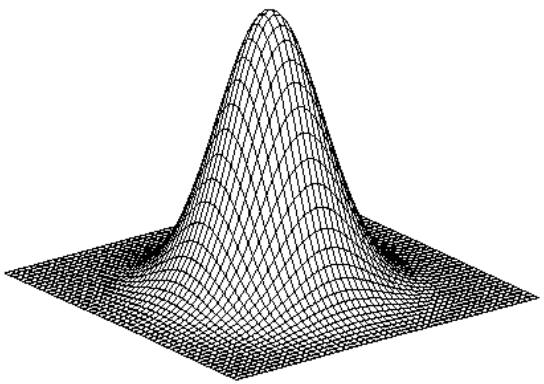


Recap: Derivatives and Edges…



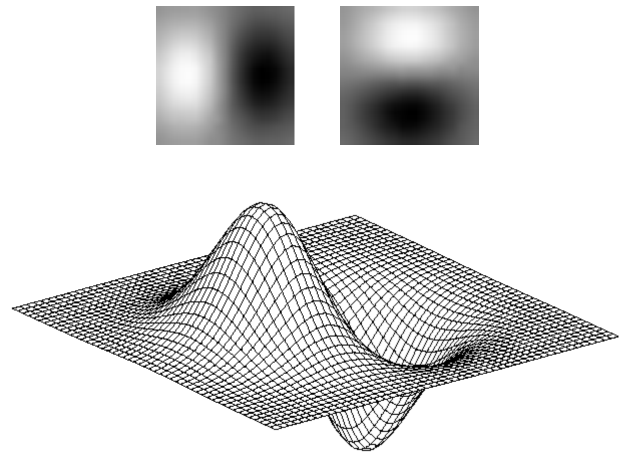
Recap: 2D Edge Detection Filters

see Exercise 2.5!



Gaussian

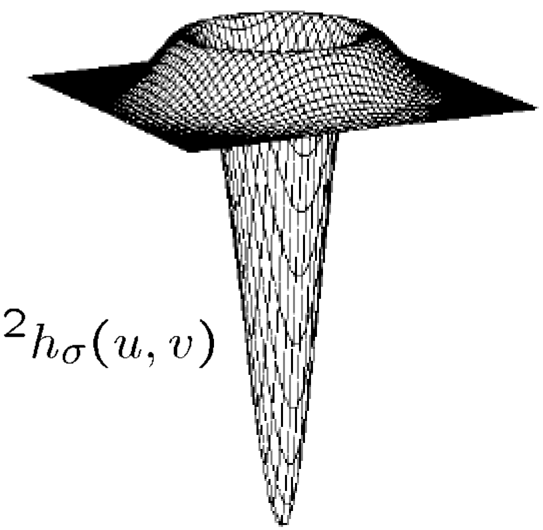
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



Derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



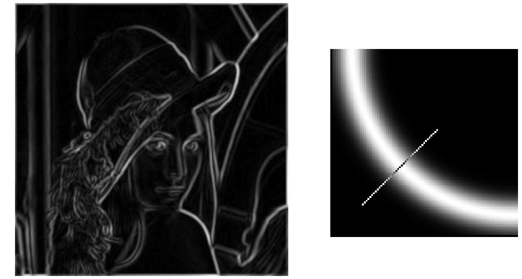
$$\nabla^2 h_\sigma(u,v)$$

- ∇^2 **is the Laplacian operator:**

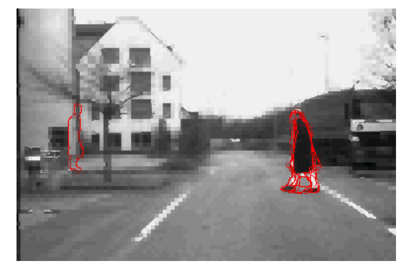
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Repetition

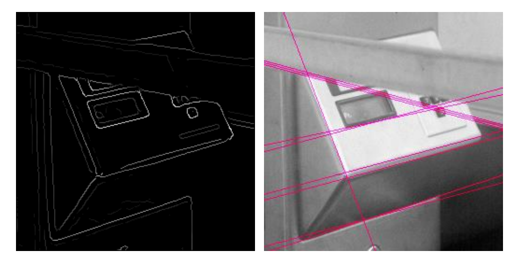
- Image Processing Basics
  - Image Formation
  - Binary Image Processing
  - Linear Filters
  - Edge & Structure Extraction
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking



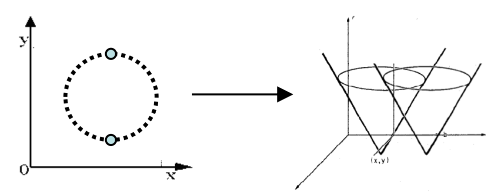
Canny edge detector



Chamfer matching



Hough transform for lines



Hough transform for circles

Recap: Canny Edge Detector

see Exercise 2.6!

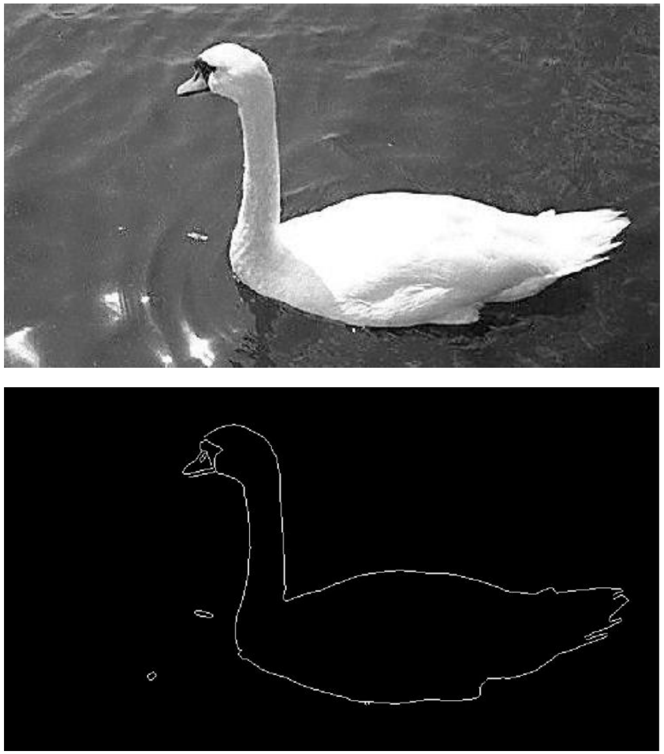
1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
4. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

- MATLAB:

```
>> edge(image,'canny');
>> help edge
```

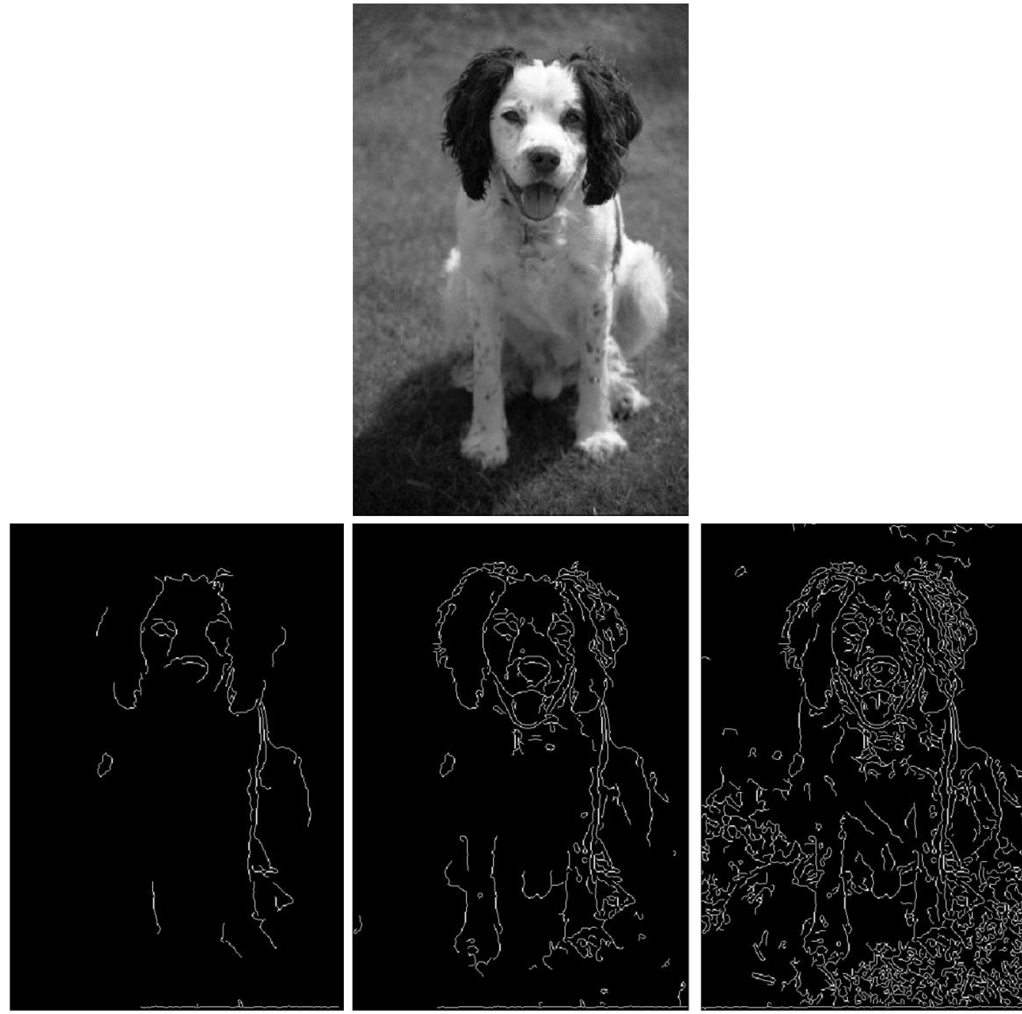
adapted from D. Lowe, L. Fei-Fei

Recap: Edges vs. Boundaries



Edges useful signal to indicate occluding boundaries, shape.

Here the raw edge output is not so bad…



…but quite often boundaries of interest are fragmented, and we have extra "clutter" edge points.

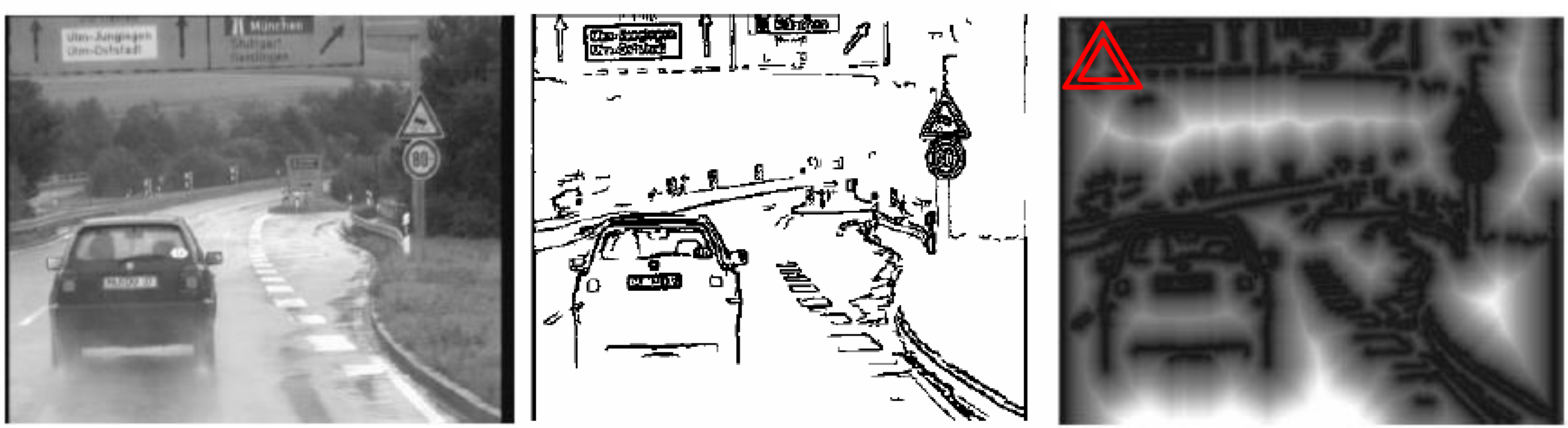
Slide credit: Kristen Grauman

Recap: Chamfer Matching

- Chamfer Distance
  - Average distance to nearest feature

$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

  - This can be computed efficiently by correlating the edge template with the distance-transformed image



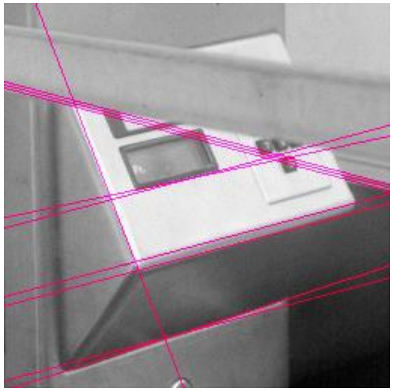
Edge image

Distance transform image

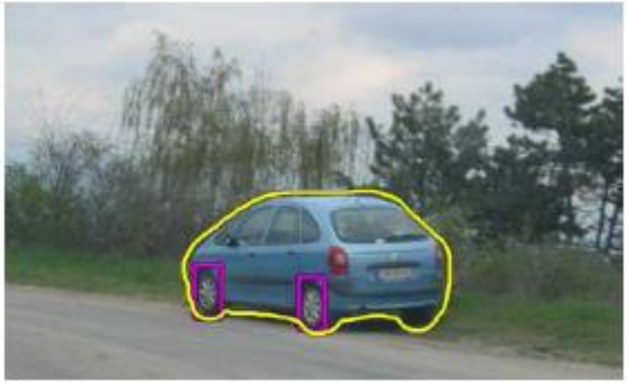
[D. Gavrila, DAGM'99]

RWTH AACHEN UNIVERSITY

Recap: Fitting and Hough Transform



Given a model of interest, we can overcome some of the missing and noisy edges using fitting techniques.



With voting methods like the Hough transform, detected points vote on possible model parameters.

Slide credit: Kristen Grauman

Recap: Hough Transform

see Exercise 3.1!

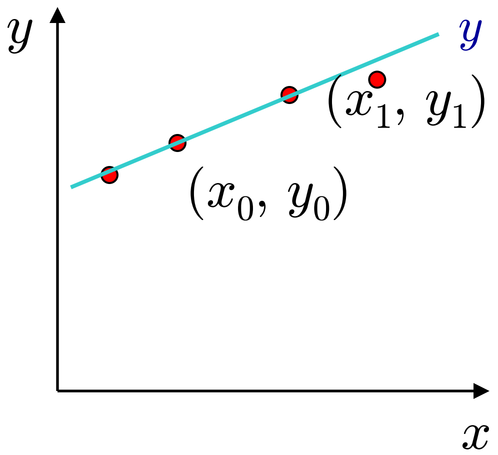
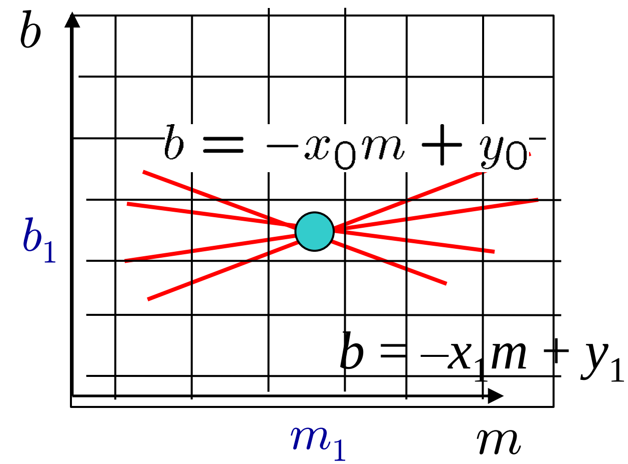


Image space



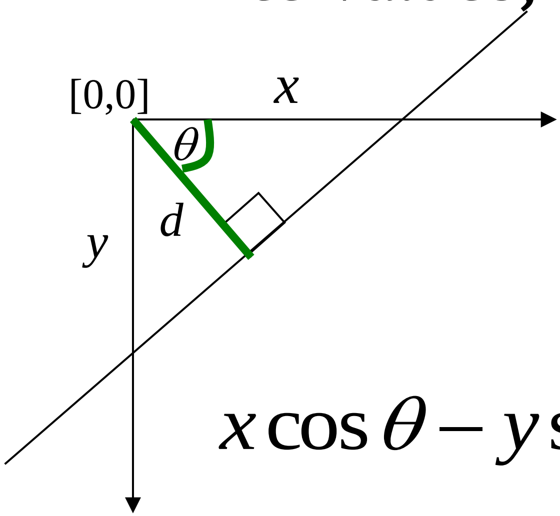
Hough (parameter) space

- How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?
  - Let each edge point in image space *vote* for a set of possible parameters in Hough space
  - Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.

Slide credit: Steve Seitz

Recap: Hough Transf. Polar Parametrization

- Usual (m,b) parameter space problematic: can take on infinite values, undefined for vertical lines.

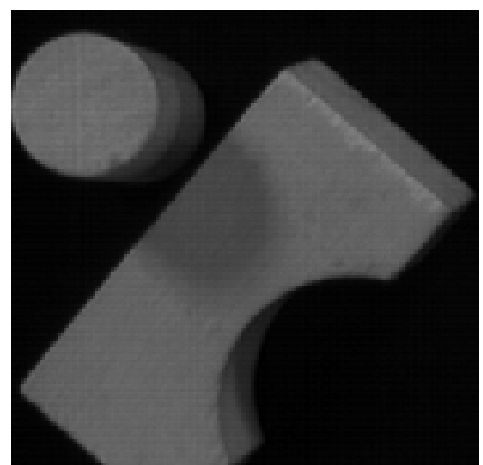


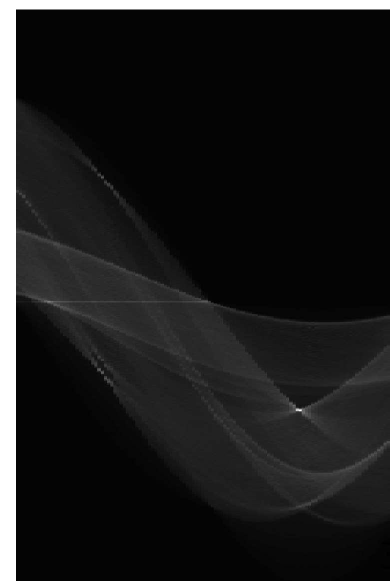
$$x\cos\theta - y\sin\theta = d$$

d : perpendicular distance from line to origin

θ : angle the perpendicular makes with the x-axis

- Point in image space
  $\Rightarrow$ sinusoid segment in
  Hough space





Slide credit: Steve Seitz

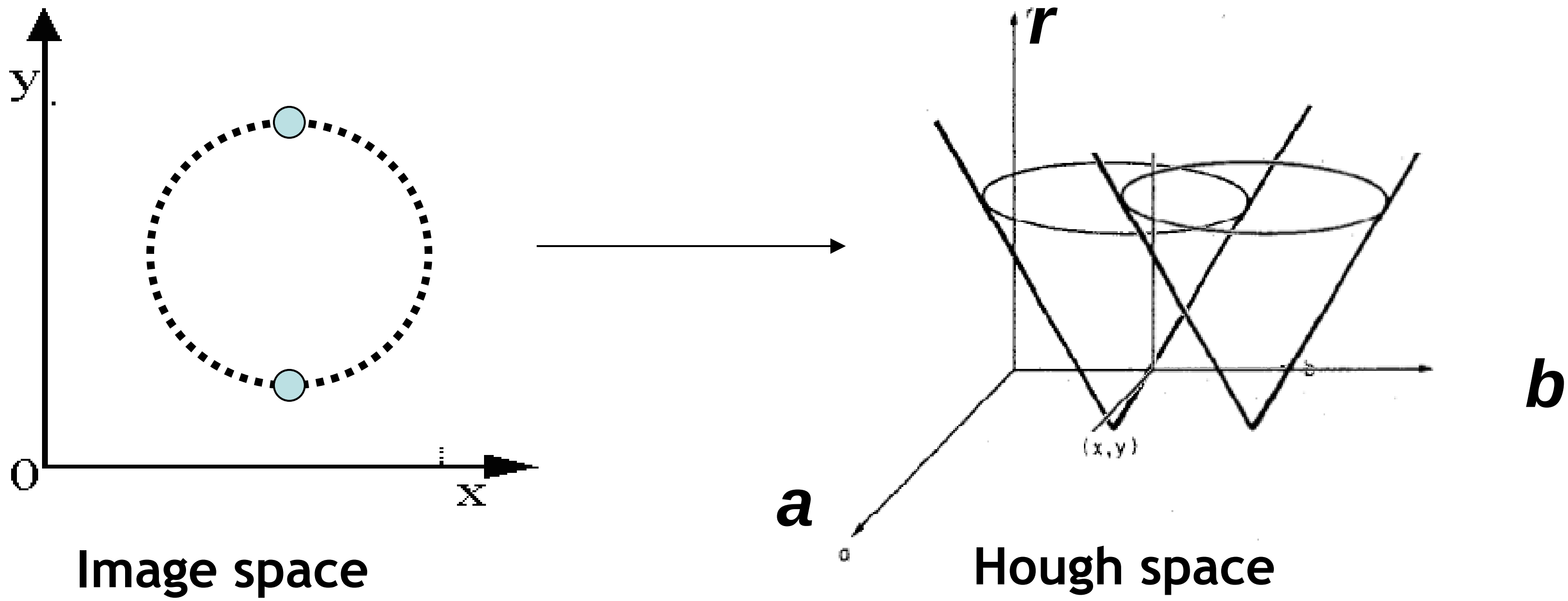
Recap: Hough Transform for Circles

see Exercise 3.1!

- **Circle: center (a,b) and radius r**

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- **For an unknown radius r, unknown gradient direction**



Slide credit: Kristen Grauman

Recap: Generalized Hough Transform

- What if want to detect arbitrary shapes defined by boundary points and a reference point?

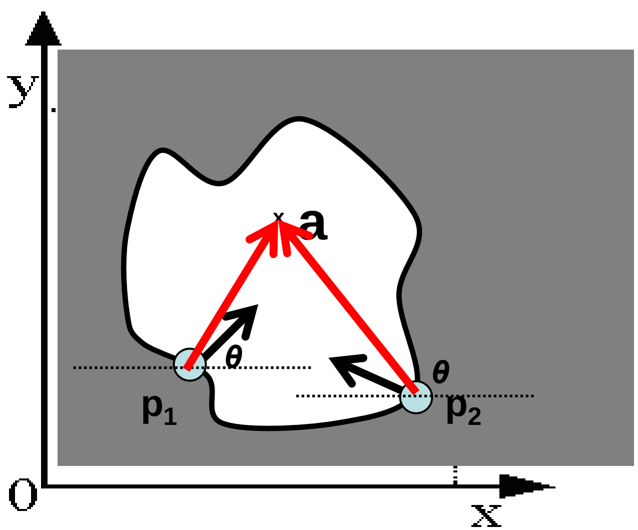


Image space

At each boundary point, compute displacement vector: $r = a - p_i$.

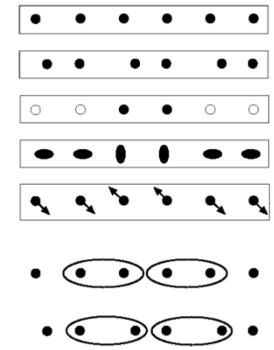
For a given model shape: store these vectors in a table indexed by gradient orientation θ.

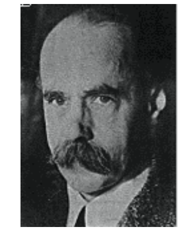
[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

Slide credit: Kristen Grauman

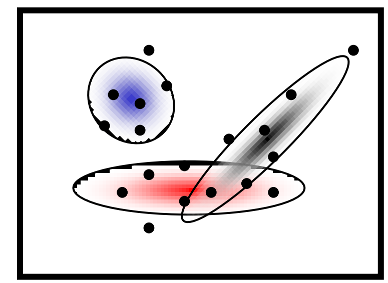
Repetition

- Image Processing Basics
- Segmentation & Grouping
  - Segmentation and Grouping
  - Segmentation as Energy Minimization
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

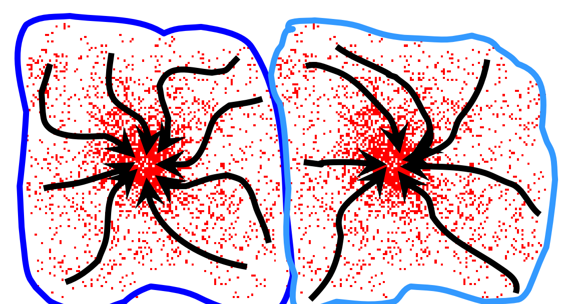




Gestalt factors



K-Means & EM clustering



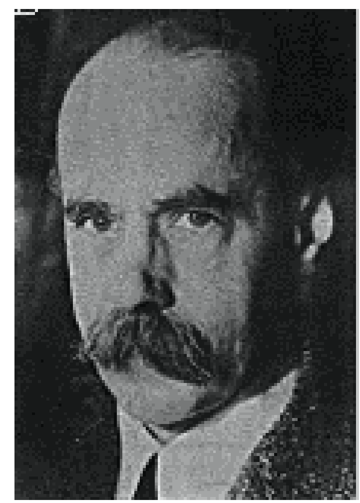
Mean-shift clustering

Recap: Gestalt Theory

- Gestalt: whole or group
  - Whole is greater than sum of its parts
  - Relationships among parts can yield new properties/features
- Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

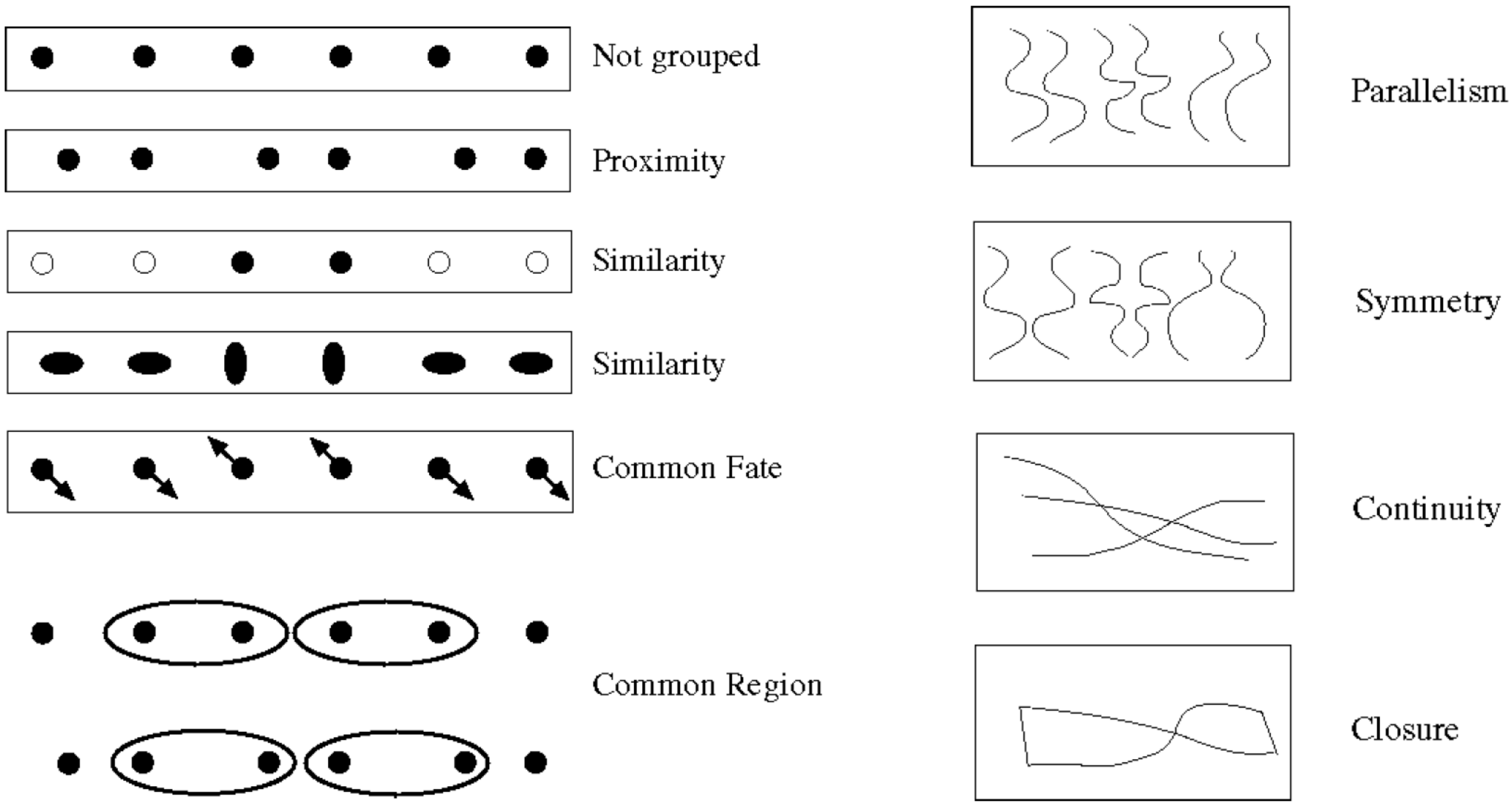
"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees."

Max Wertheimer
(1880-1943)



Untersuchungen zur Lehre von der Gestalt,
Psychologische Forschung, Vol. 4, pp. 301-350, 1923
http://psy.ed.asu.edu/~classics/Wertheimer/Forms/forms.htm

Recap: Gestalt Factors

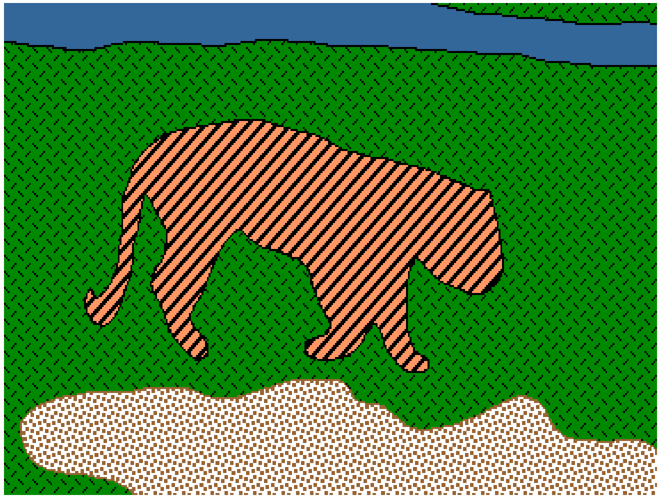


- **These factors make intuitive sense, but are very difficult to translate into algorithms.**

Image source: Forsyth & Ponce

Recap: Image Segmentation

- Goal: identify groups of pixels that go together



Slide credit: Steve Seitz, Kristen Grauman

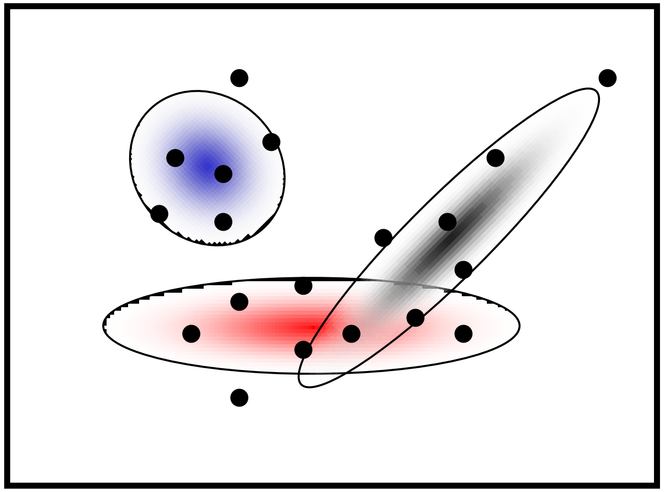
Recap: K-Means Clustering

- **Basic idea: randomly initialize the *k* cluster centers, and iterate between the two following steps**
  1. Randomly initialize the cluster centers, $c_1, \ldots, c_K$
  2. Given cluster centers, determine points in each cluster
     - For each point p, find the closest c_i. Put p into cluster i
  3. Given points in each cluster, solve for c_i
     - Set c_i to be the mean of points in cluster i
  4. If c_i have changed, repeat Step 2

- **Properties**
  - Will always converge to *some* solution
  - Can be a "local minimum"
    - Does not always find the global minimum of objective function:

$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} \|p - c_i\|^2$$

Recap: Expectation Maximization (EM)



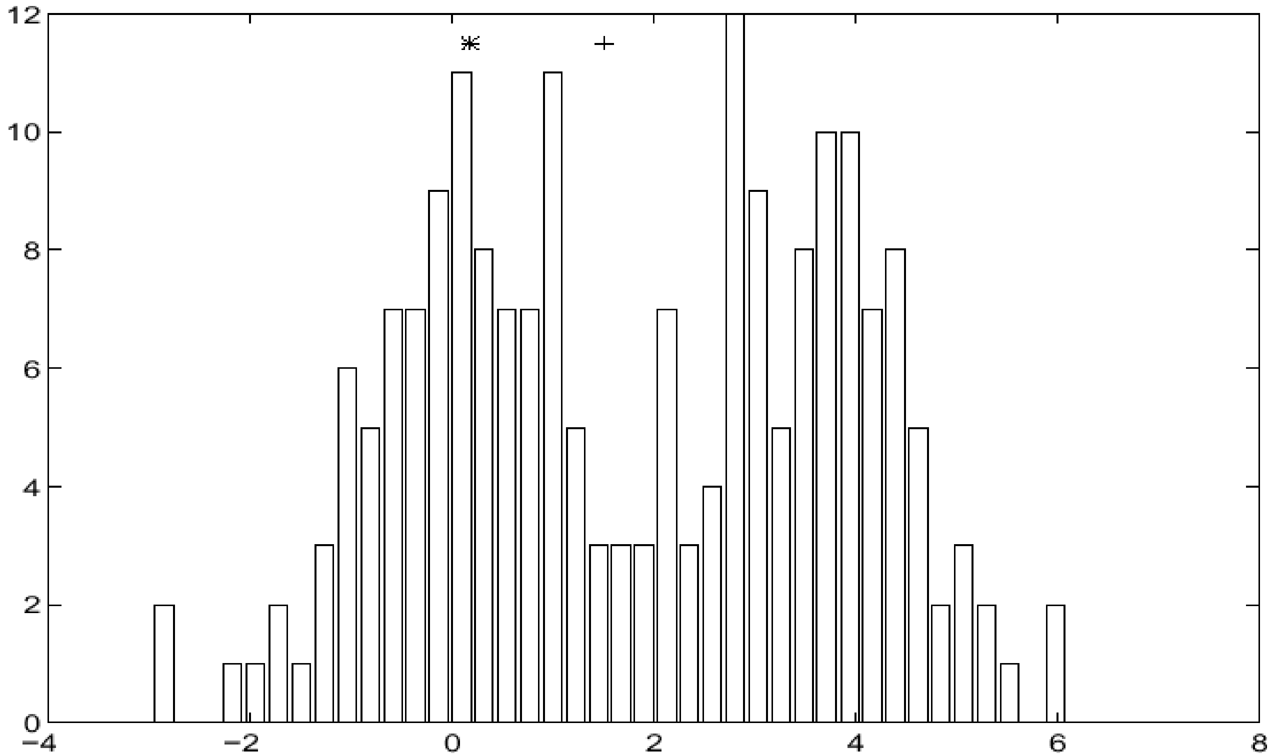
- Goal
  - Find blob parameters θ that maximize the likelihood function:

$$p(data|\theta) = \prod_{n=1}^{N} p(\mathbf{x}_n|\theta)$$

- Approach:
  1. E-step: given current guess of blobs, compute ownership of each point
  2. M-step: given ownership probabilities, update blobs to maximize likelihood function
  3. Repeat until convergence

Recap: Mean-Shift Algorithm

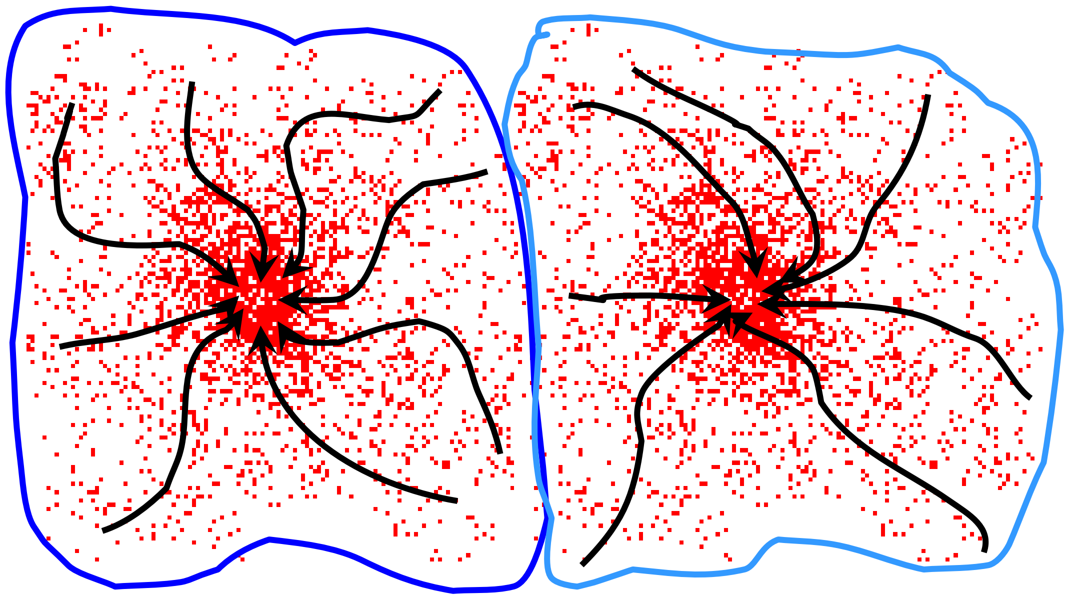
see Exercise 3.2!



- **Iterative Mode Search**
  1. Initialize random seed, and window W
  2. Calculate center of gravity (the "mean") of W: $\sum_{x \in W} x H(x)$
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence

Recap: Mean-Shift Clustering

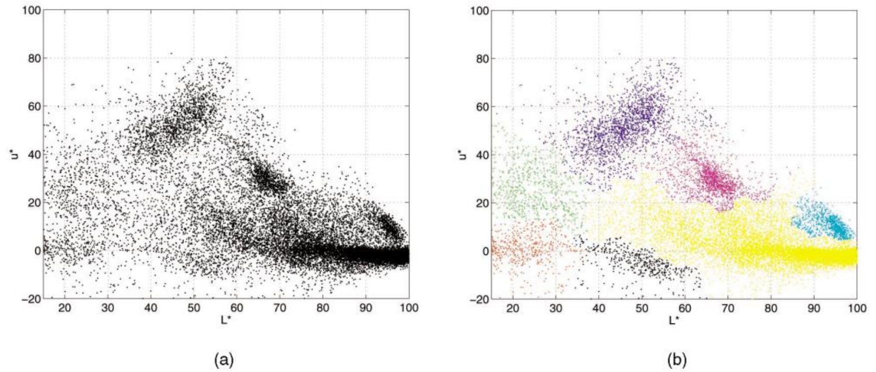
- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode

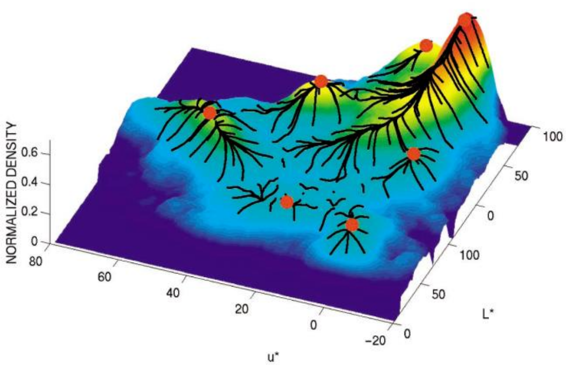


Recap: Mean-Shift Segmentation

see Exercise 3.3!

- Find features (color, gradients, texture, etc)
- Initialize windows at individual pixel locations
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode

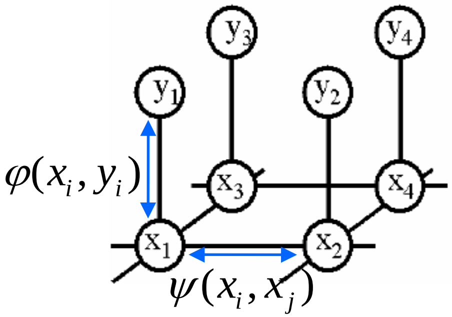




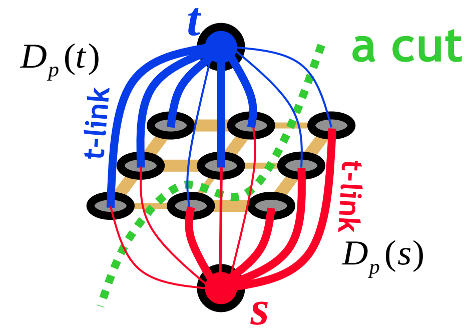
Slide credit: Svetlana Lazebnik

Repetition

- Image Processing Basics
- Segmentation & Grouping
  - Segmentation and Grouping
  - Segmentation as Energy Minimization
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking



Markov Random Fields

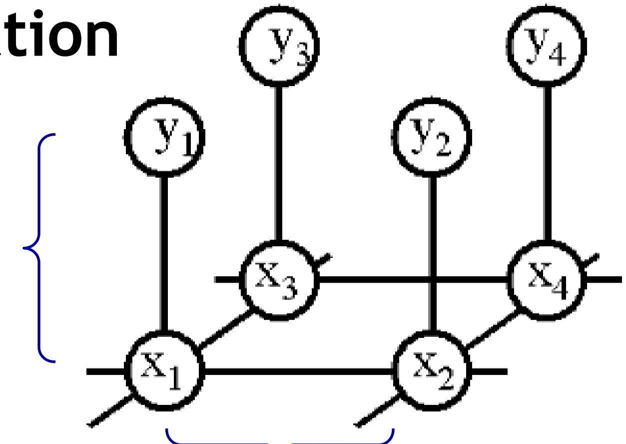


Graph cuts

Recap: MRFs for Image Segmentation

- **MRF formulation**

Unary potentials
$\phi(x_i, y_i)$



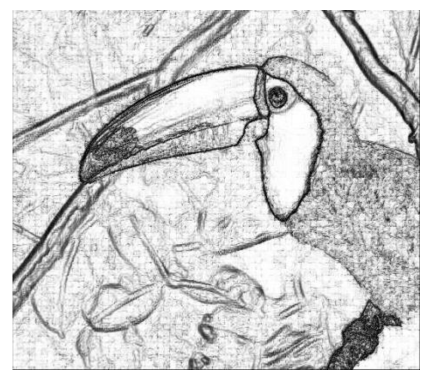
Pairwise potentials
$\psi(x_i, x_j)$

⇒ Minimize the energy

$$E(\mathbf{x}, \mathbf{y}) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j)$$

Data (D)

Unary likelihood

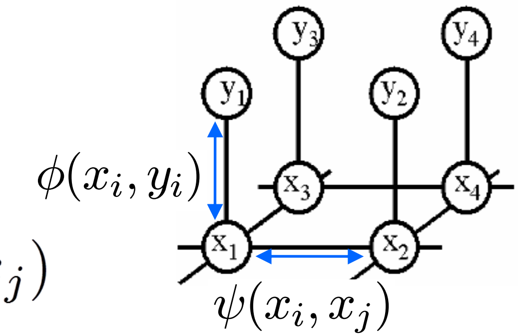


Pair-wise Terms

MAP Solution

Slide adapted from Phil Torr

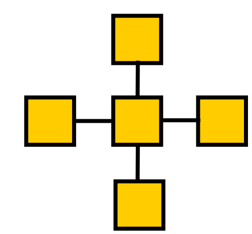
Recap: Energy Formulation



- Energy function

$$E(\mathbf{x}, \mathbf{y}) = \underbrace{\sum_i \phi(x_i, y_i)}_{\text{Unary potentials}} + \underbrace{\sum_{i,j} \psi(x_i, x_j)}_{\text{Pairwise potentials}}$$

- Unary potentials ϕ
  - Encode local information about the given pixel/patch
  - How likely is a pixel/patch to belong to a certain class (e.g. foreground/background)?
- Pairwise potentials ψ
  - Encode neighborhood information
  - How different is a pixel/patch's label from that of its neighbor? (e.g. based on intensity/color/texture difference, edges)

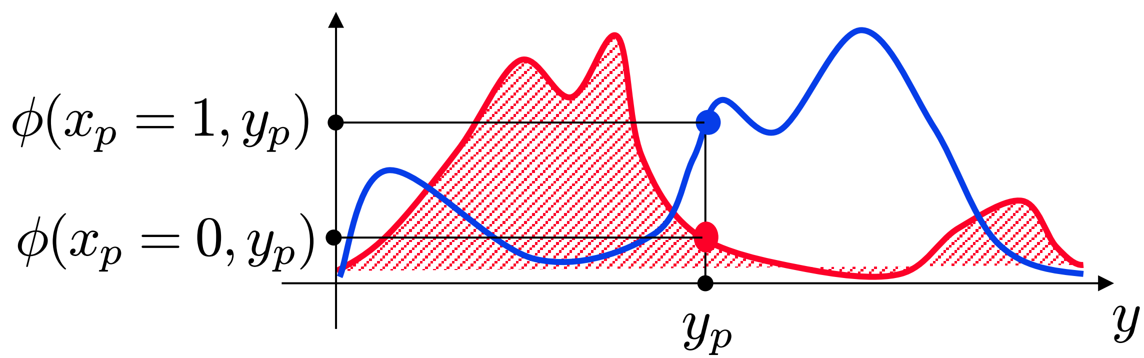


Recap: How to Set the Potentials?

- **Unary potentials**
  - **E.g. color model, modeled with a Mixture of Gaussians**

$$\phi(x_i, y_i; \theta_\phi) = \log \sum_k \theta_\phi(x_i, k) p(k|x_i) \mathcal{N}(y_i; \bar{y}_k, \Sigma_k)$$

$\Rightarrow$ **Learn color distributions for each label**



Recap: How to Set the Potentials?

- Pairwise potentials
  - Potts Model

$$\psi(x_i, x_j; \theta_\psi) = \theta_\psi \delta(x_i \neq x_j)$$

    - Simplest discontinuity preserving model.
    - Discontinuities between any pair of labels are penalized equally.
    - Useful when labels are unordered or number of labels is small.

  - Extension: "Contrast sensitive Potts model"

$$\psi(x_i, x_j, g_{ij}(y); \theta_\psi) = \theta_\psi g_{ij}(y) \delta(x_i \neq x_j)$$

where

$$g_{ij}(y) = e^{-\beta \|y_i - y_j\|^2} \qquad \beta = 2 / avg\left(\|y_i - y_j\|^2\right)$$

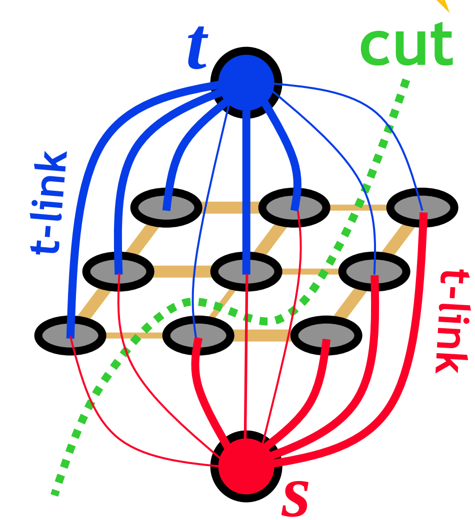
$\Rightarrow$ Discourages label changes except in places where there is also a large change in the observations.

Recap: Graph-Cuts Energy Minimization

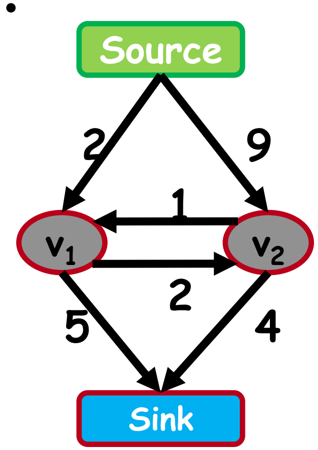
see Exercise 3.4!

- Solve an equivalent graph cut problem
  1. Introduce extra nodes: source and sink
  2. Weight connections to source/sink (t-links) by $\phi(x_i = s)$ and $\phi(x_i = t)$, respectively.
  3. Weight connections between nodes (n-links) by $\psi(x_i, x_j)$.
  4. Find the minimum cost cut that separates source from sink.

  ⇒ Solution is equivalent to minimum of the energy.



- s-t Mincut can be solved efficiently
  - Dual to the well-known max flow problem
  - Very efficient algorithms available for regular grid graphs (1-2 MPixels/s)
  - Globally optimal result for 2-class problems



Recap: When Can s-t Graph Cuts Be Applied?

Unary potentials   Pairwise potentials

$$E(L) = \sum_{p} E_p(L_p) + \sum_{pq \in N} E(L_p, L_q)$$

t-links   n-links

$$L_p \in \{s, t\}$$

- s-t graph cuts can only globally minimize **binary energies** that are **submodular**. [Boros & Hummer, 2002, Kolmogorov & Zabih, 2004]

E(L) can be minimized by *s-t* graph cuts $\Leftrightarrow$ $E(s,s)+E(t,t) \leq E(s,t)+E(t,s)$

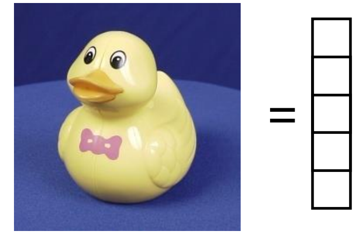
Submodularity ("convexity")

- Submodularity is the discrete equivalent to convexity.
  - Implies that every local energy minimum is a global minimum.

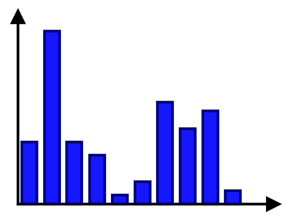
  $\Rightarrow$ Solution will be globally optimal.

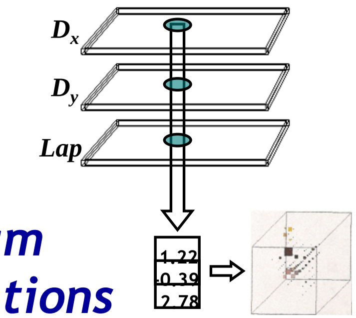
Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
  - Global Representations
  - Subspace Representations
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

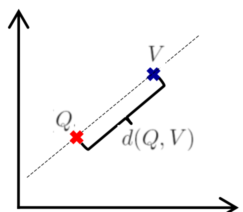


Appearance-based recognition





Histogram representations

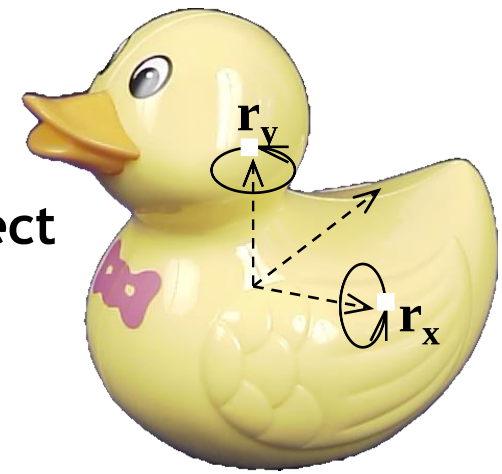


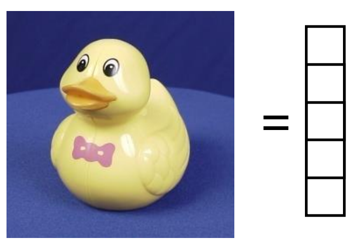
Comparison measures

Recap: Appearance-Based Recognition

- Basic assumption
  - Objects can be represented by a set of images ("appearances").
  - For recognition, it is sufficient to just compare the 2D appearances.
  - No 3D model is needed.

3D object

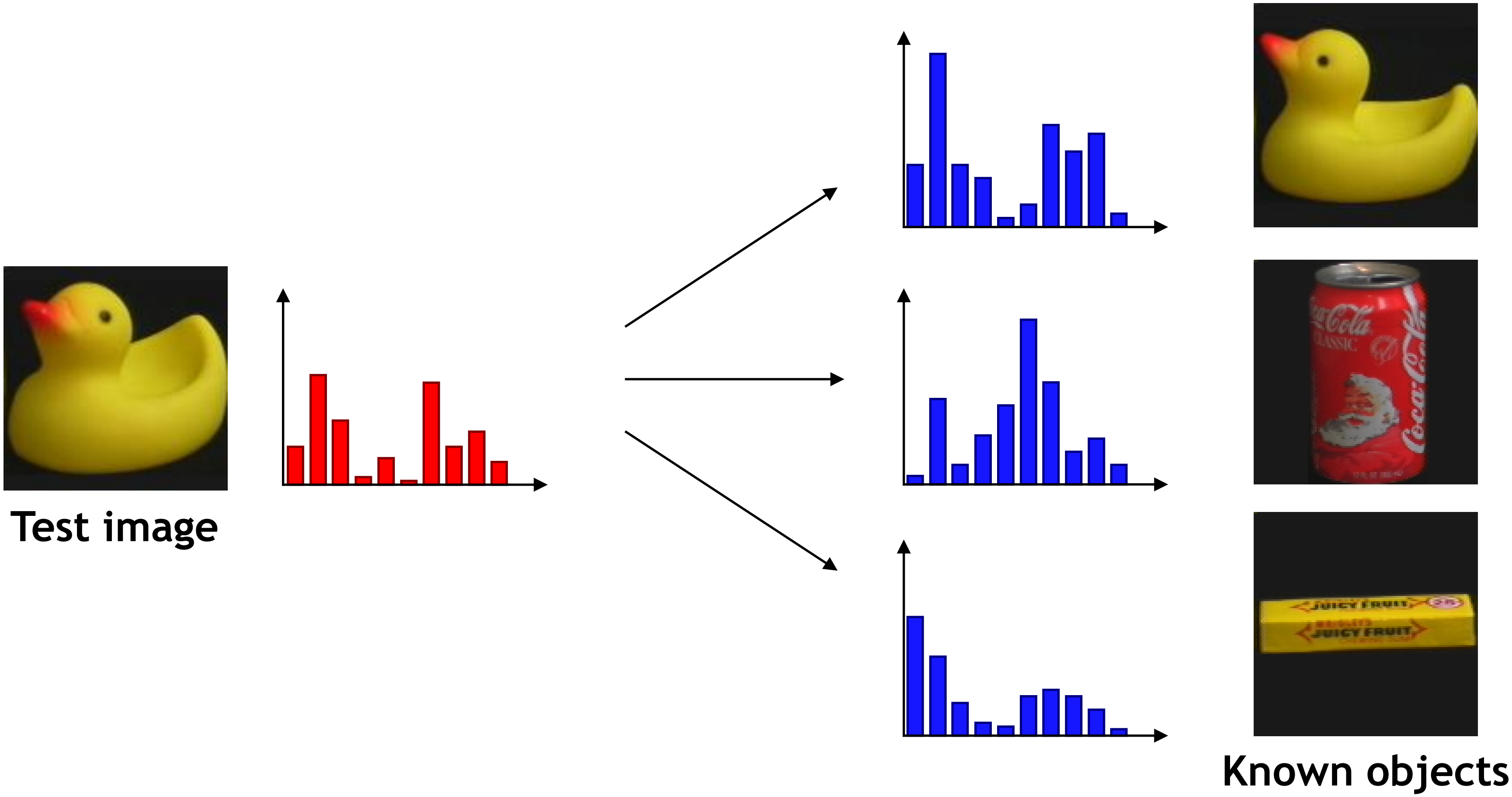




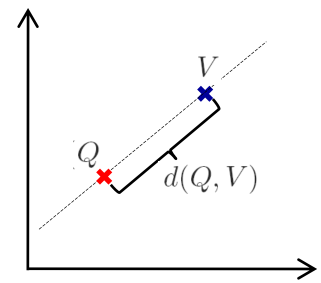
⇒ **Fundamental paradigm shift in the 90's**

Recap: Recognition Using Global Features

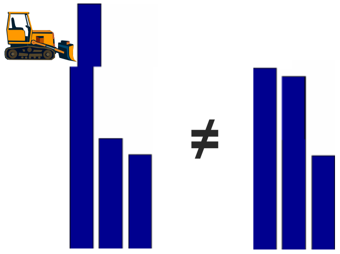
- E.g. histogram comparison



Recap: Comparison Measures



- **Vector space interpretation**
  - Euclidean distance
- **Statistical motivation**
  - Chi-square
  - Bhattacharyya
- **Information-theoretic motivation**
  - Kullback-Leibler divergence, Jeffreys divergence
- **Histogram motivation**
  - Histogram intersection
- **Ground distance**
  - Earth Movers Distance (EMD)



Recap: Recognition Using Histograms

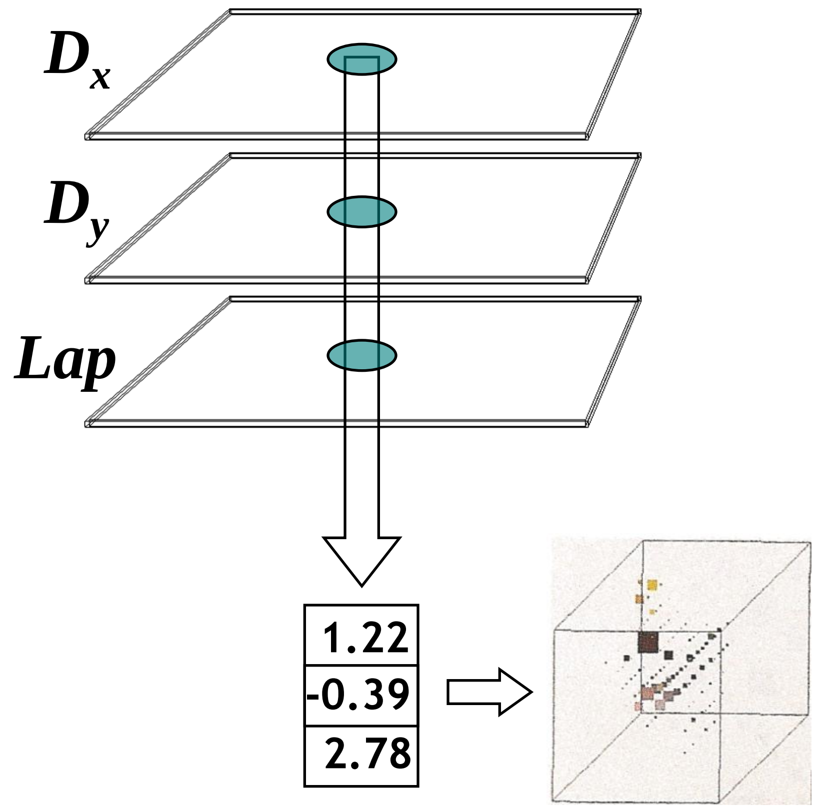
see Exercise 4.2!

- Simple algorithm
  1. Build a set of histograms $H=\{h_i\}$ for each known object
     - More exactly, for each *view* of each object
  2. Build a histogram h_t for the test image.
  3. Compare h_t to each $h_i \in H$
     - Using a suitable comparison measure
  4. Select the object with the best matching score
     - Or reject the test image if no object is similar enough.

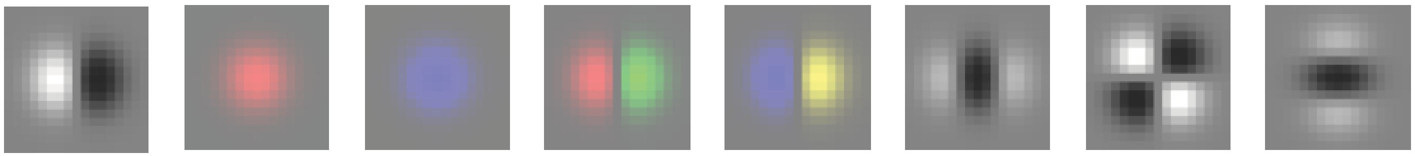
"Nearest-Neighbor" strategy

Recap: Multidimensional Representations

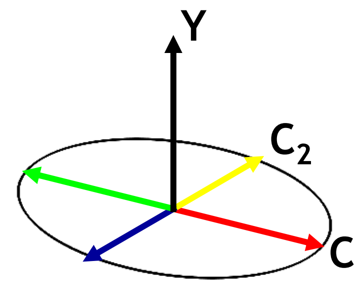
- Combination of several descriptors
  - Each descriptor is applied to the whole image.
  - Corresponding pixel values are combined into one feature vector.
  - Feature vectors are collected in multidimensional histogram.



Recap: Colored Derivatives



- **Generalization: derivatives along**
  - Y axis $\rightarrow$ intensity differences
  - C_1 axis $\rightarrow$ red-green differences
  - C_2 axis $\rightarrow$ blue-yellow differences



- **Application:**
  - Brand identification in video

[Hall & Crowley, 2000]

RWTH AACHEN UNIVERSITY

First Applications Take Up Shape…

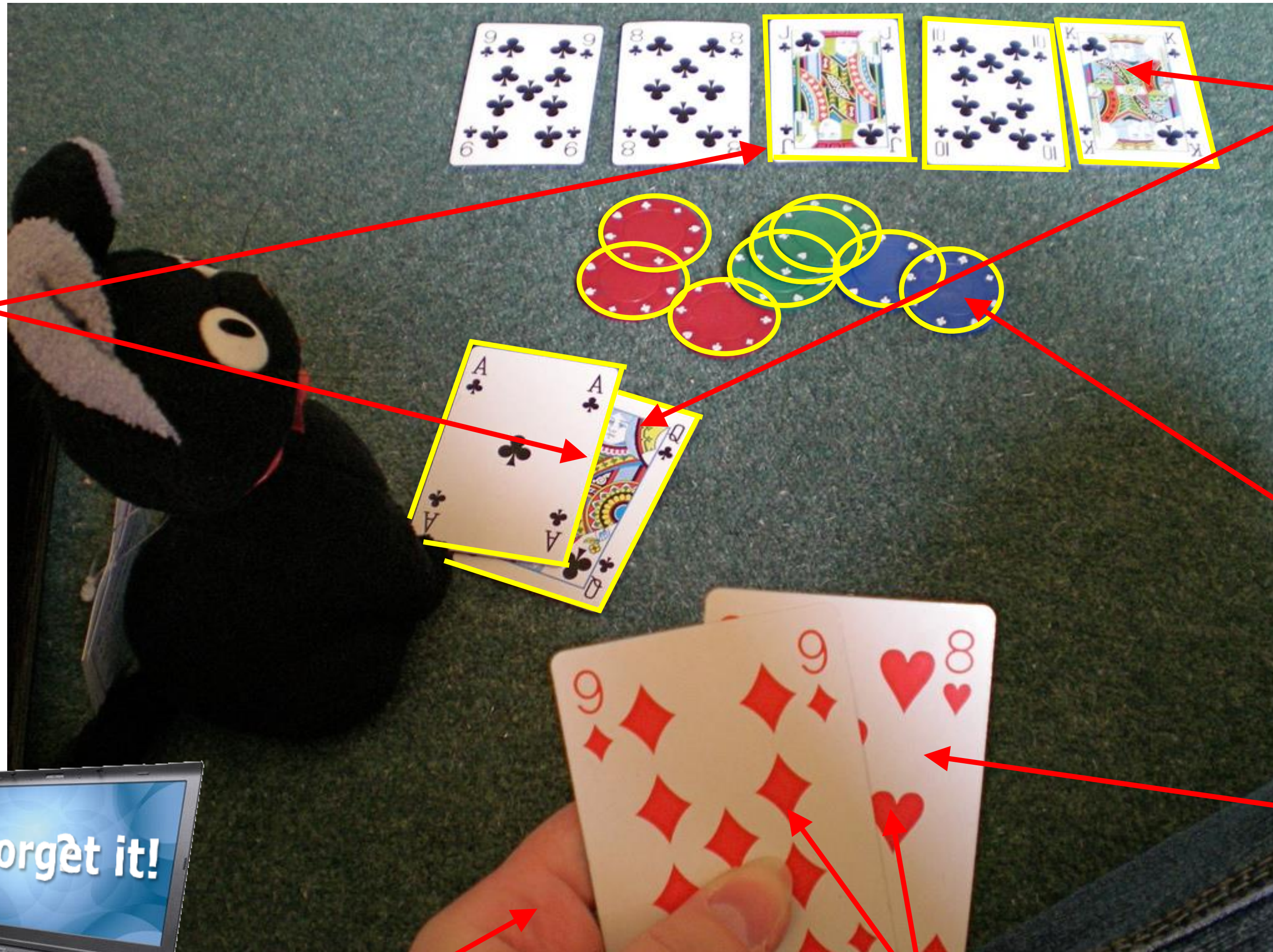
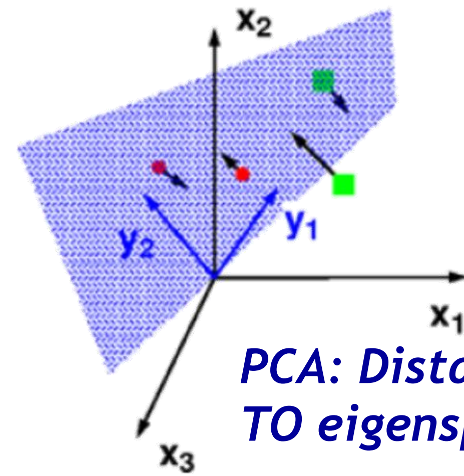


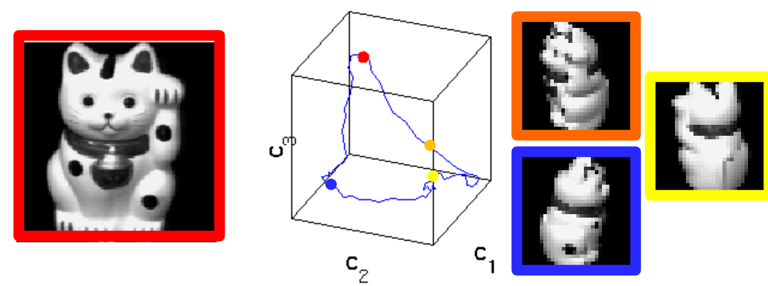
Image Source: http://www.flickr.com/photos/angelsk/2806412807/

Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
  - Global Representations
  - Subspace Representations
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

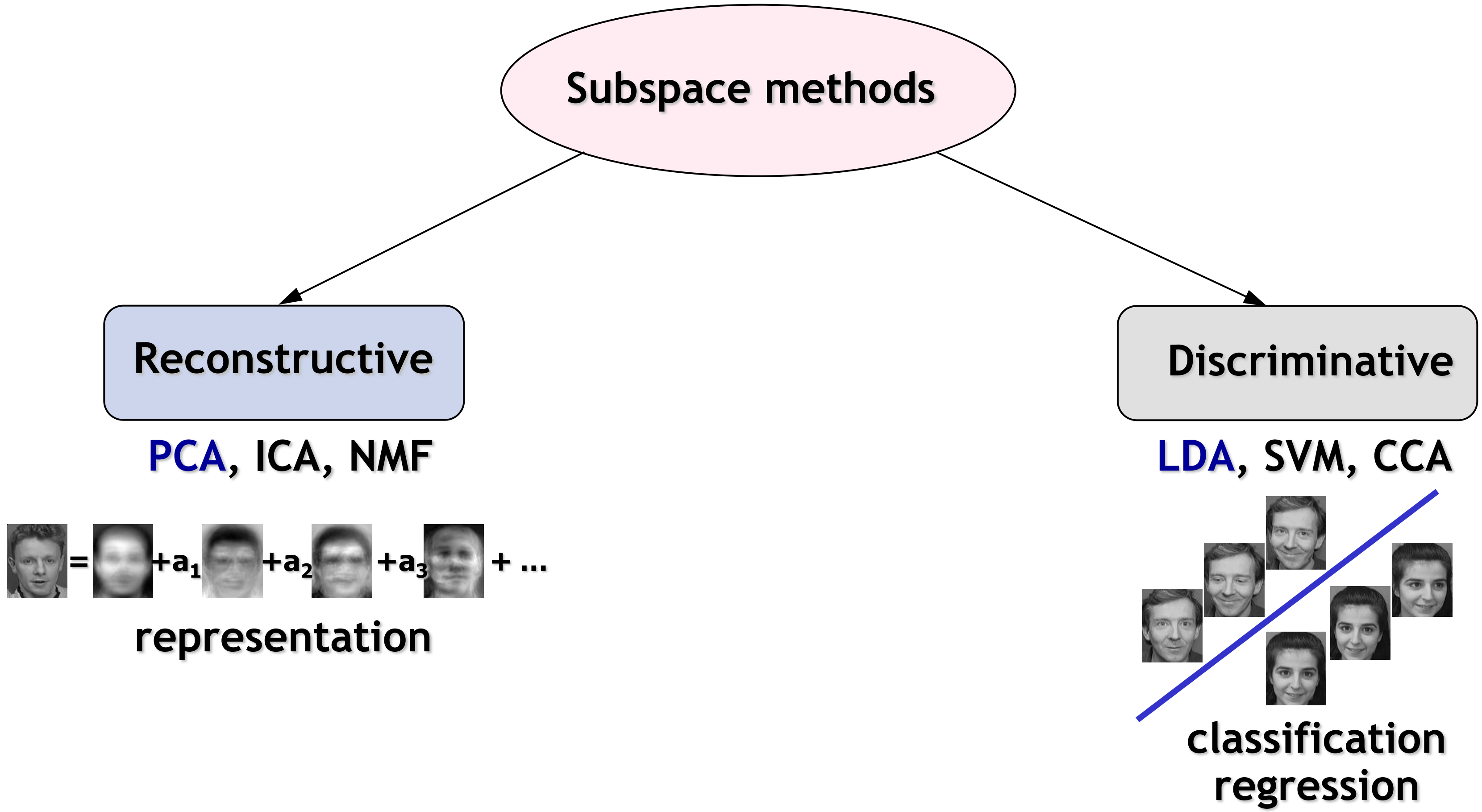


PCA: Distance TO eigenspace



PCA: Distance IN eigenspace

Recap: Subspace Methods



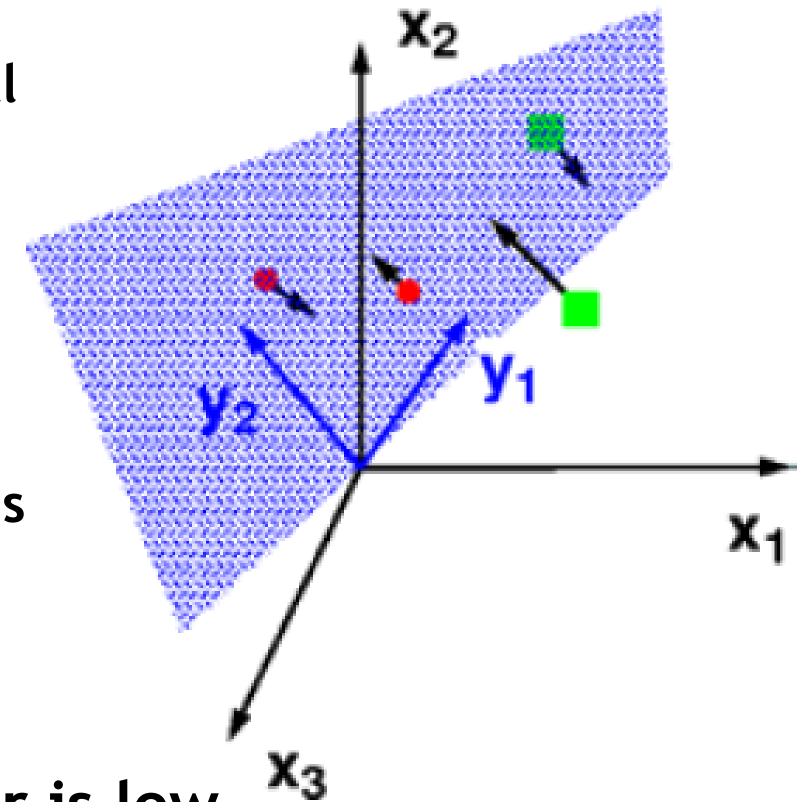
Recap: Obj. Detection by Distance TO Eigenspace

- For each test image, compute the reprojection error
  - An n-pixel image $x \in \mathbb{R}^n$ can be projected to the low-dimensional feature space $y \in \mathbb{R}^m$ by

$$y = Ux$$

  - From $y \in \mathbb{R}^m$, the reconstruction of the point is $\underline{U}^T y$
  - The error of the reconstruction is

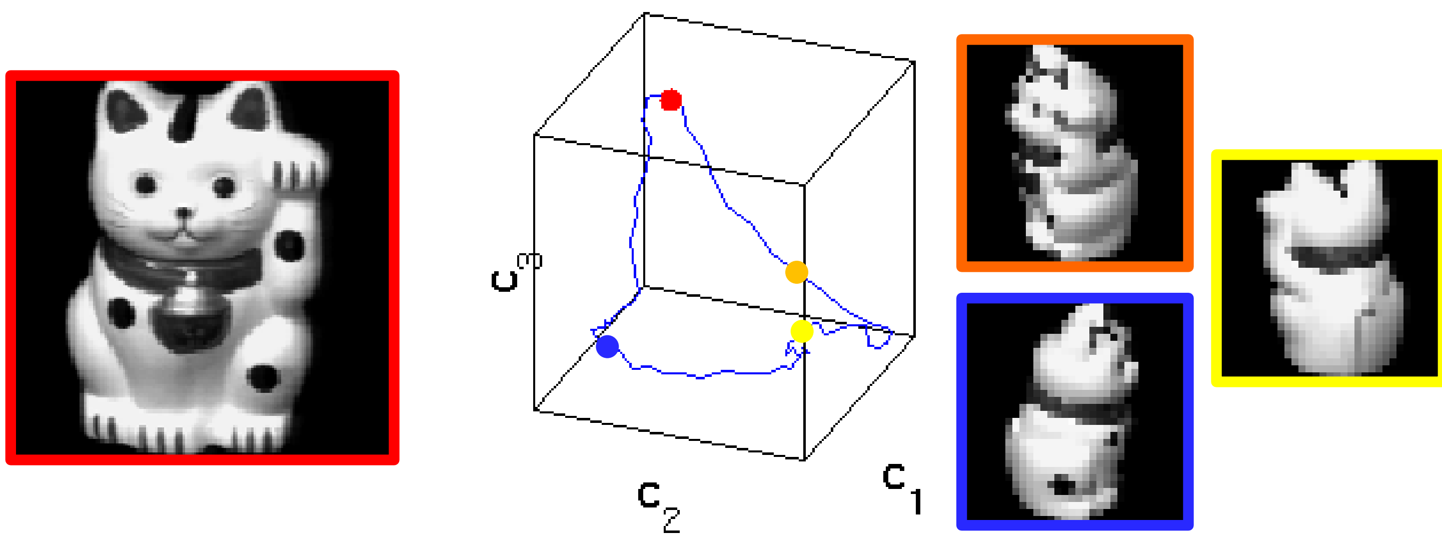
$$\left\| x - U^T U x \right\|$$



- Accept a detection if this error is low.
  - Assumption: subspace is optimized to the target object (class).
  - Other classes are not represented well ⇒ large error.

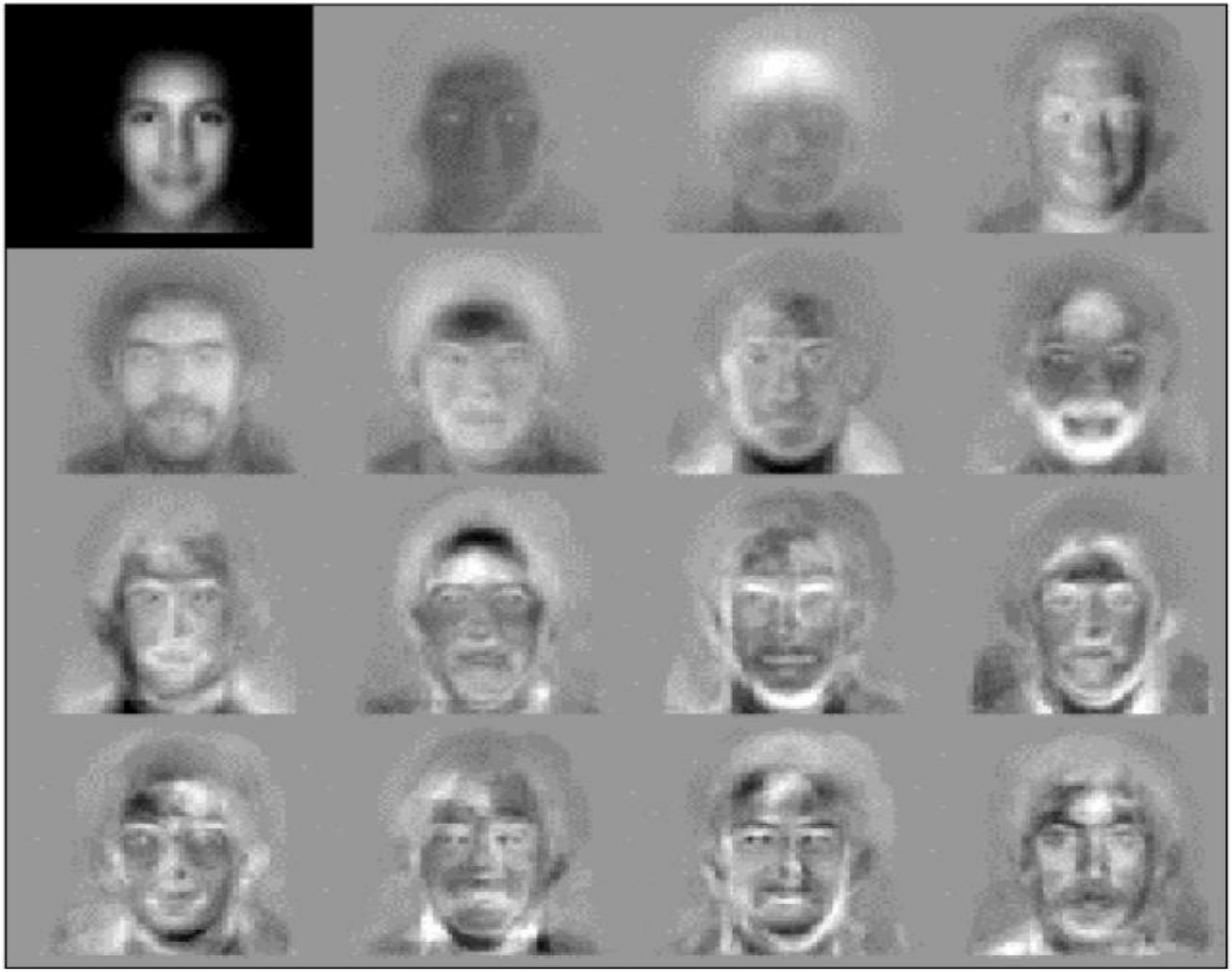
Recap: Obj Identification by Distance IN Eigenspace

- Objects are represented as coordinates in an n-dim. eigenspace.
- Example:
  - 3D space with points representing individual objects or a manifold representing parametric eigenspace (*e.g.*, orientation, pose, illumination).



- Estimate parameters by finding the NN in the eigenspace

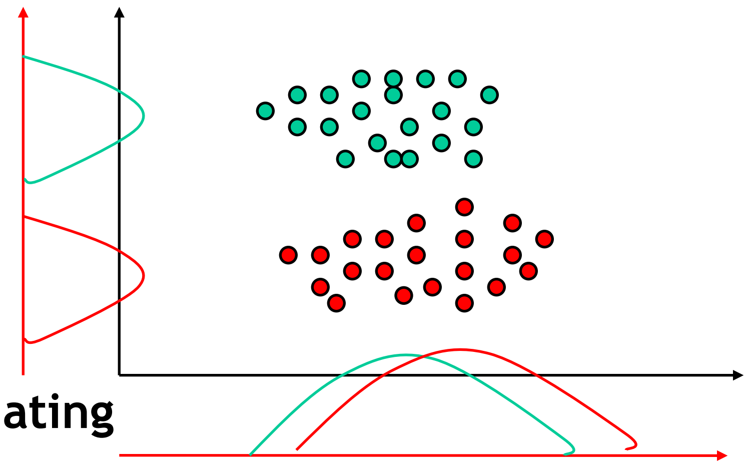
Recap: Eigenfaces



Slide credit: Peter Belhumeur

Recap: Restrictions of PCA

- PCA minimizes projection error



- PCA is „unsupervised“ no information on classes is used
- Discriminating information might be lost

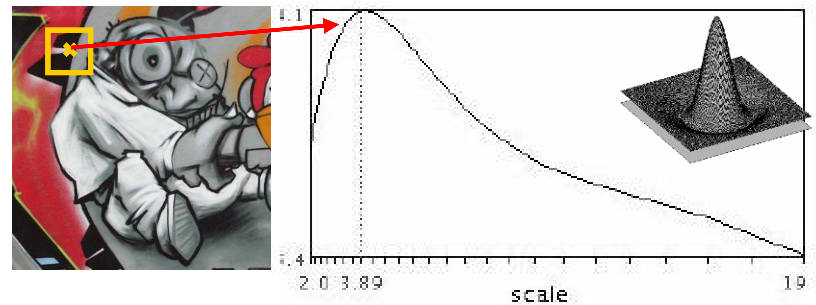
Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
  - Local Features - Detection and Description
  - Recognition with Local Features
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

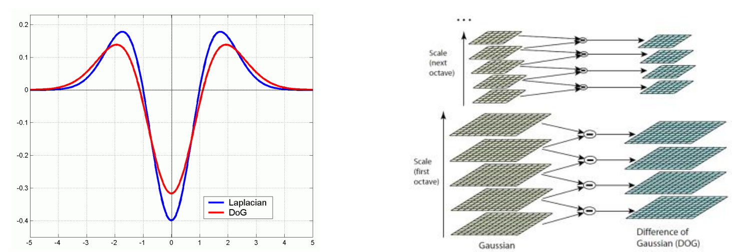
$$M(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

Harris & Hessian detector

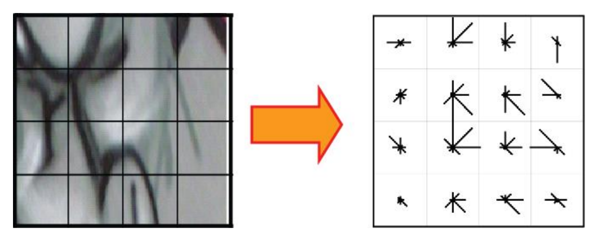
$$Hes(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$



Laplacian scale selection

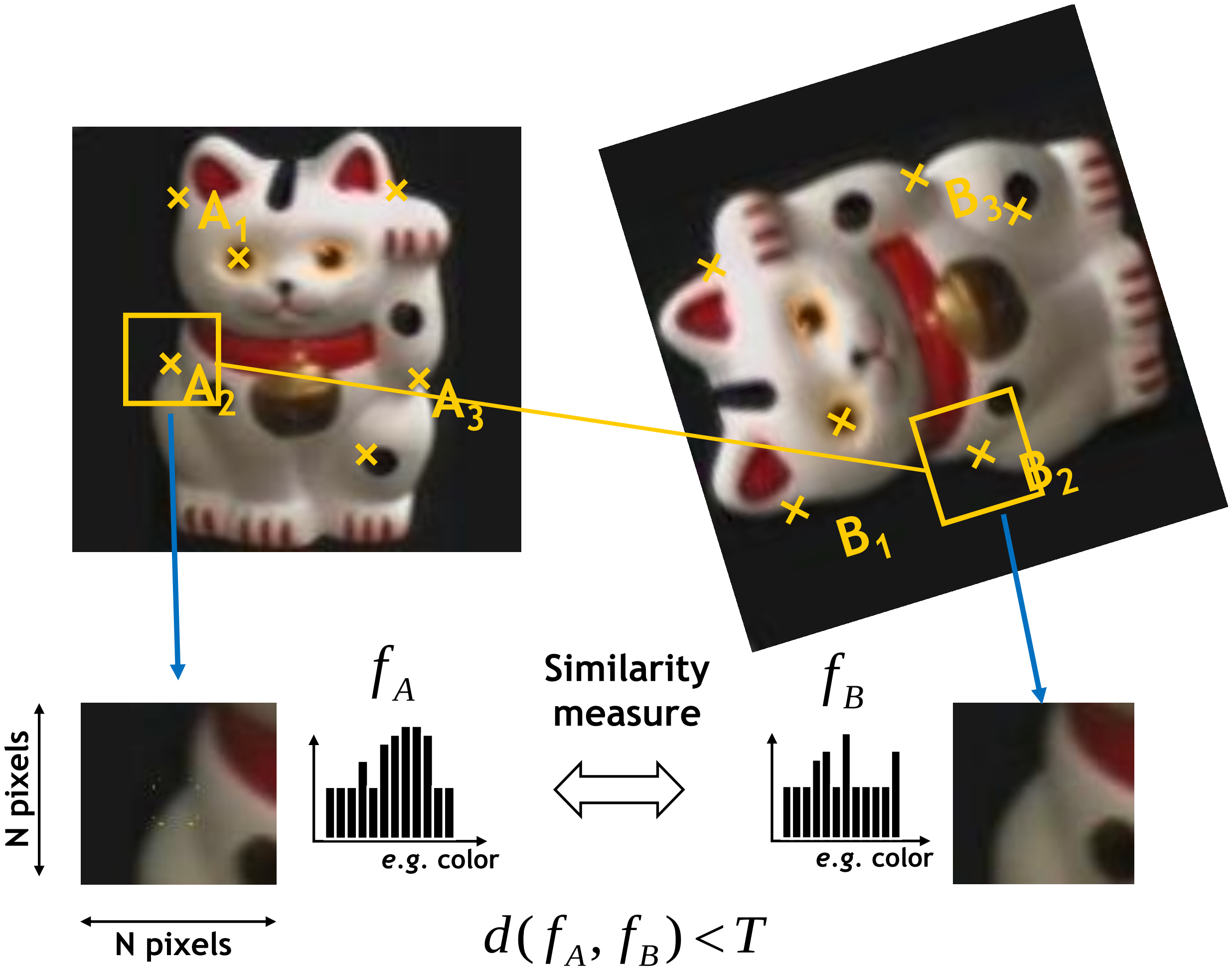


Difference-of-Gaussian (DoG)



SIFT descriptor

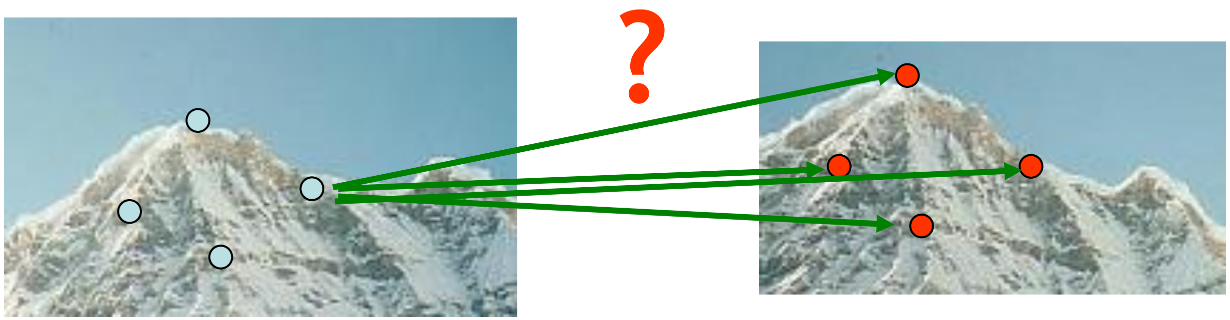
Recap: Local Feature Matching Pipeline



1. Find a set of distinctive key-points
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

Recap: Requirements for Local Features

- **Problem 1:**
  - Detect the same point *independently* in both images
- **Problem 2:**
  - For each point correctly recognize the corresponding one



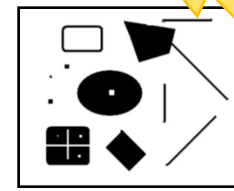
We need a repeatable detector!

We need a reliable and distinctive descriptor!

Recap: Harris Detector [Harris88]

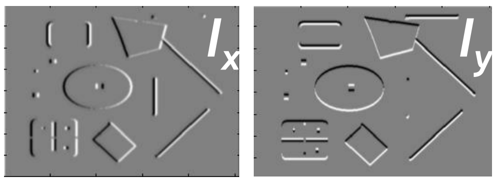
see Exercise 5.2!

- **Compute second moment matrix (autocorrelation matrix)**

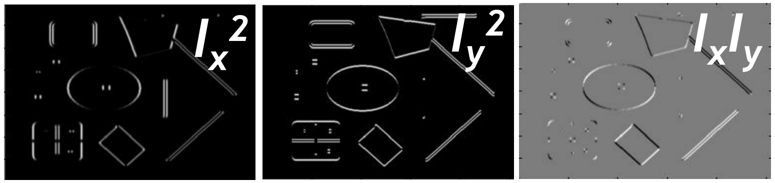


$$M(\sigma_I,\sigma_D)=g(\sigma_I)*\begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

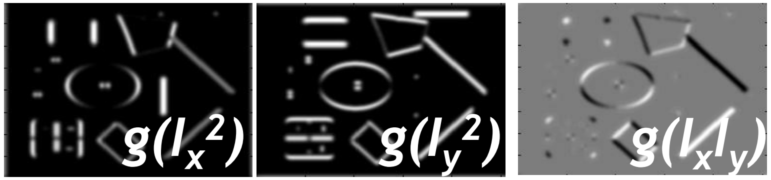
1. Image derivatives



2. Square of derivatives



3. Gaussian filter $g(\sigma_I)$



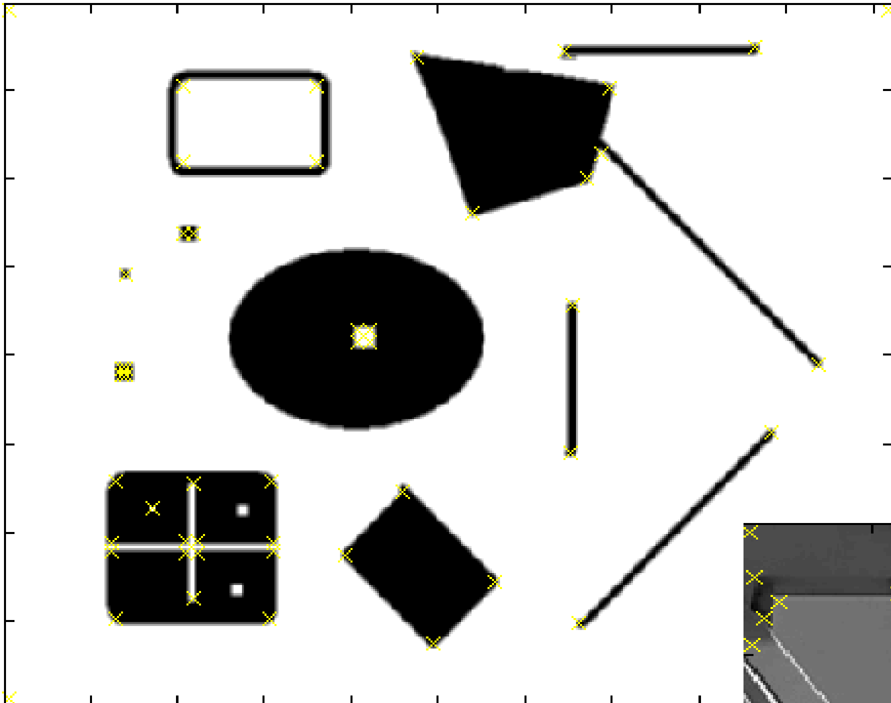
4. Cornerness function – two strong eigenvalues

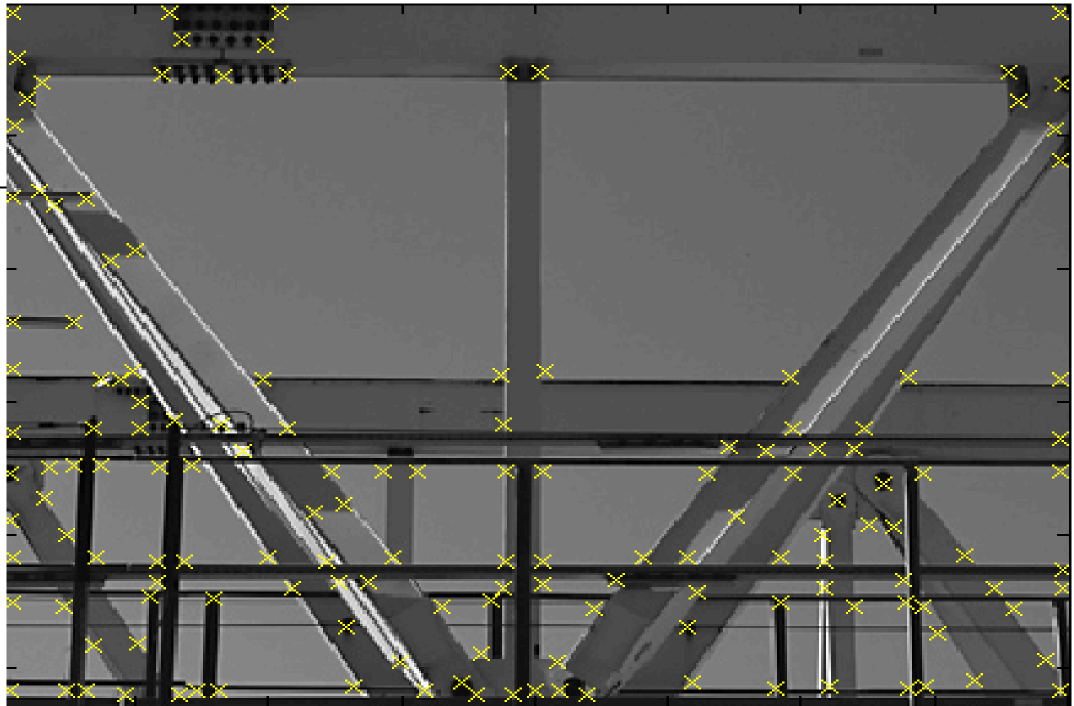
$$R=\det[M(\sigma_I,\sigma_D)]-\alpha[\operatorname{trace}(M(\sigma_I,\sigma_D))]$$

$$=g(I_x^2)g(I_y^2)-[g(I_xI_y)]^2-\alpha[g(I_x^2)+g(I_y^2)]^2$$

5. Perform non-maximum suppression

Recap: Harris Detector Responses [Harris88]





Effect: **A very precise corner detector.**

Recap: Hessian Detector [Beaudet78]

see Exercise 5.1!

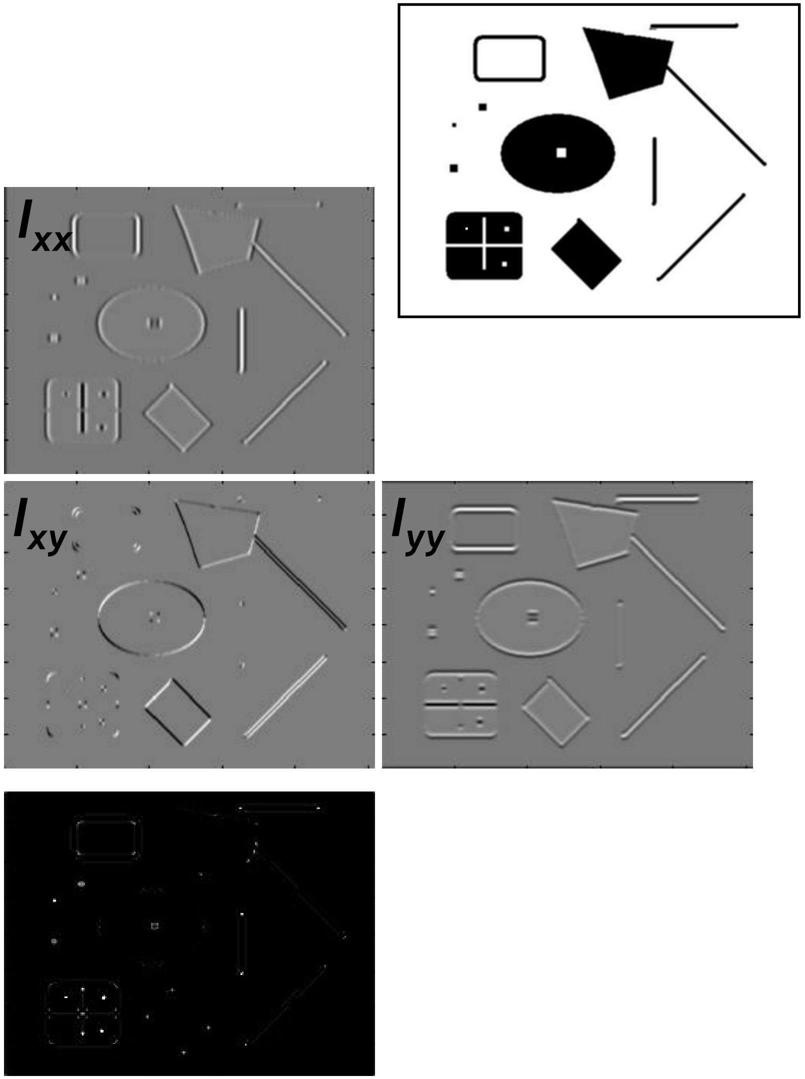
- **Hessian determinant**

$$Hessian(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

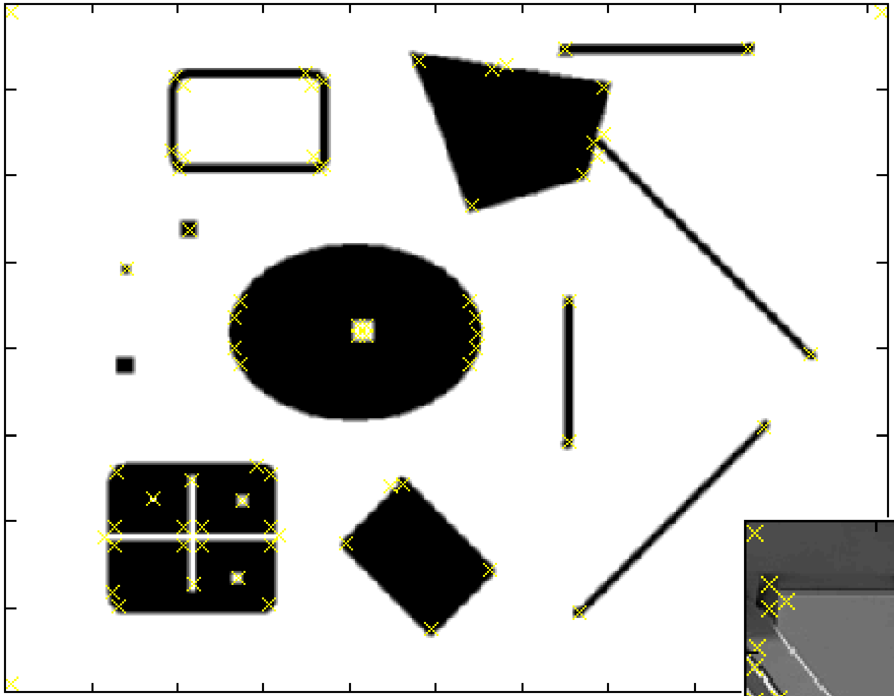
$$\det(Hessian(I)) = I_{xx} I_{yy} - I_{xy}^2$$

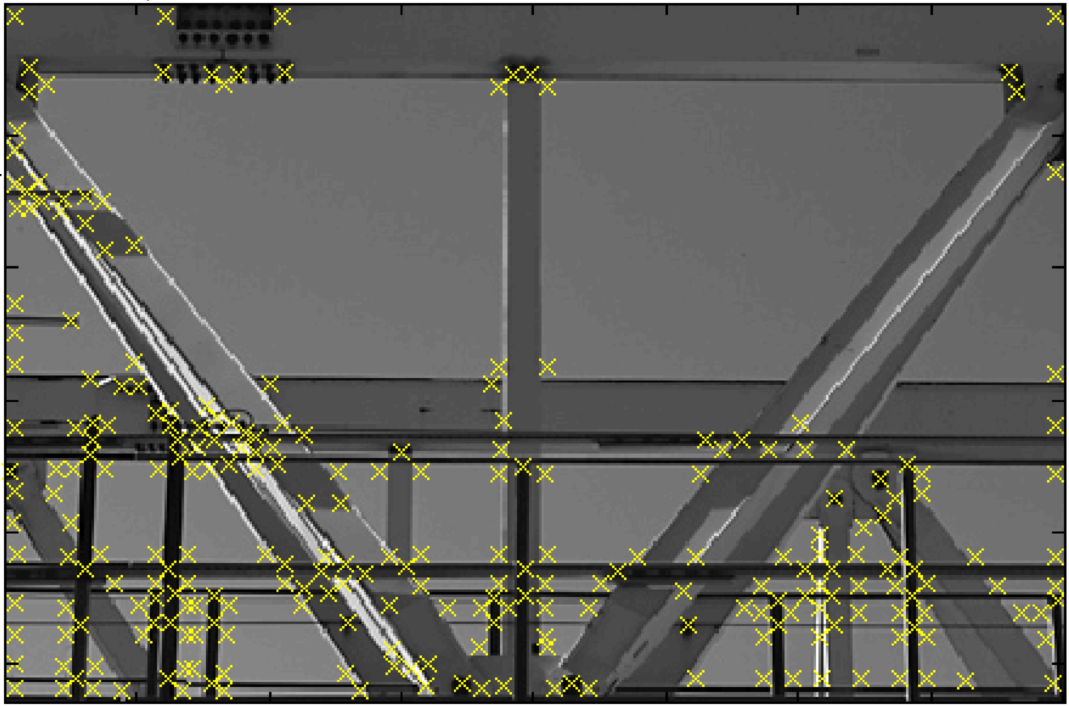
In Matlab:

$$I_{xx} .* I_{yy} - (I_{xy})^\wedge 2$$



Recap: Hessian Detector Responses [Beaudet78]



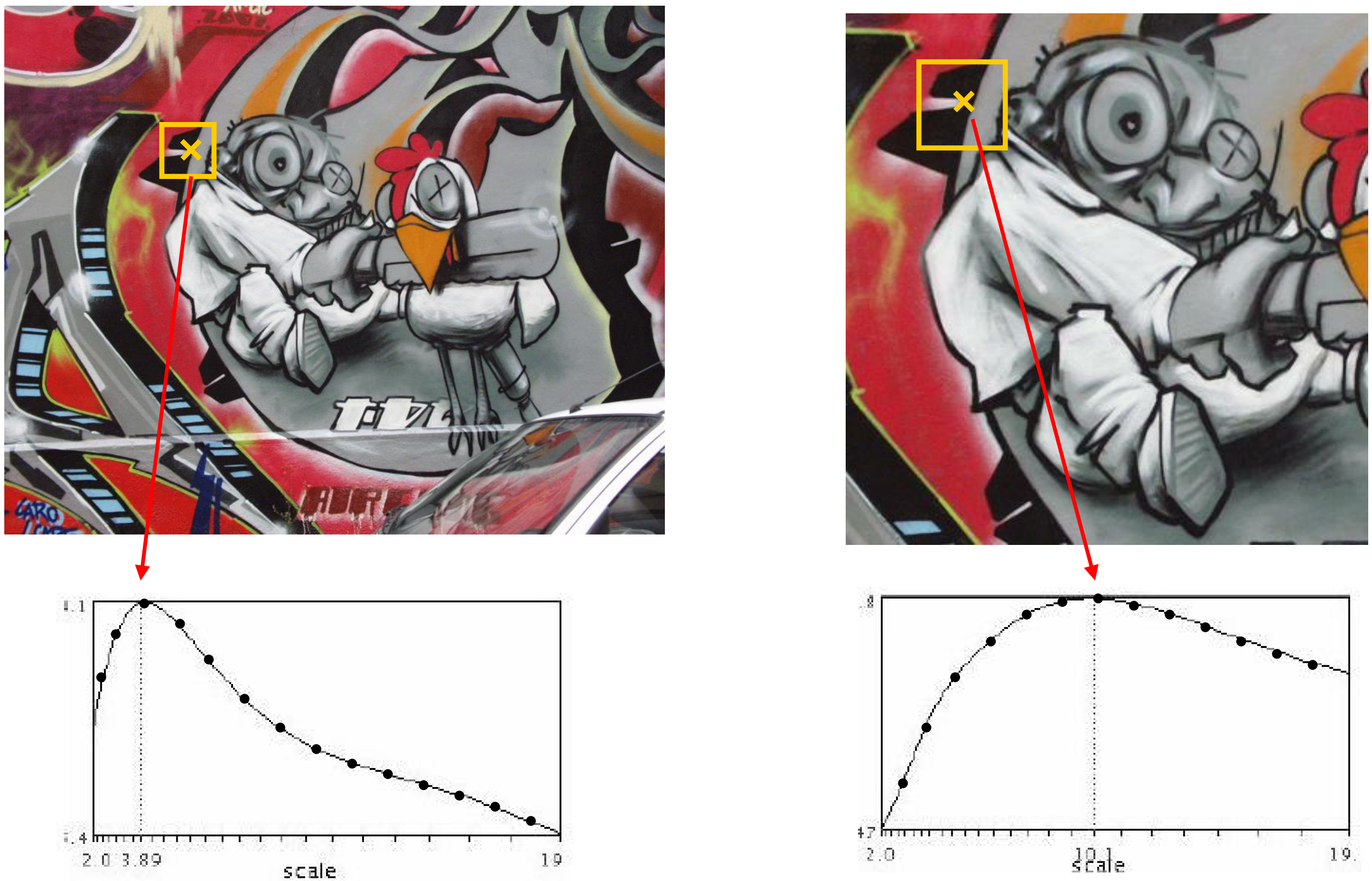


Effect: Responses mainly on corners and strongly textured areas.

Slide credit: Krystian Mikolajczyk

Recap: Automatic Scale Selection

- Function responses for increasing scale (scale signature)

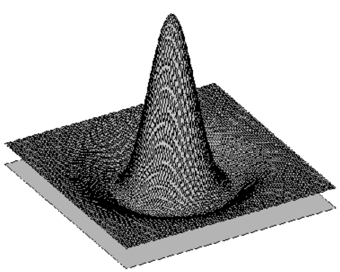


$f(I_{i_1 \ldots i_m}(x, \sigma))$ $\quad$ $f(I_{i_1 \ldots i_m}(x', \sigma'))$

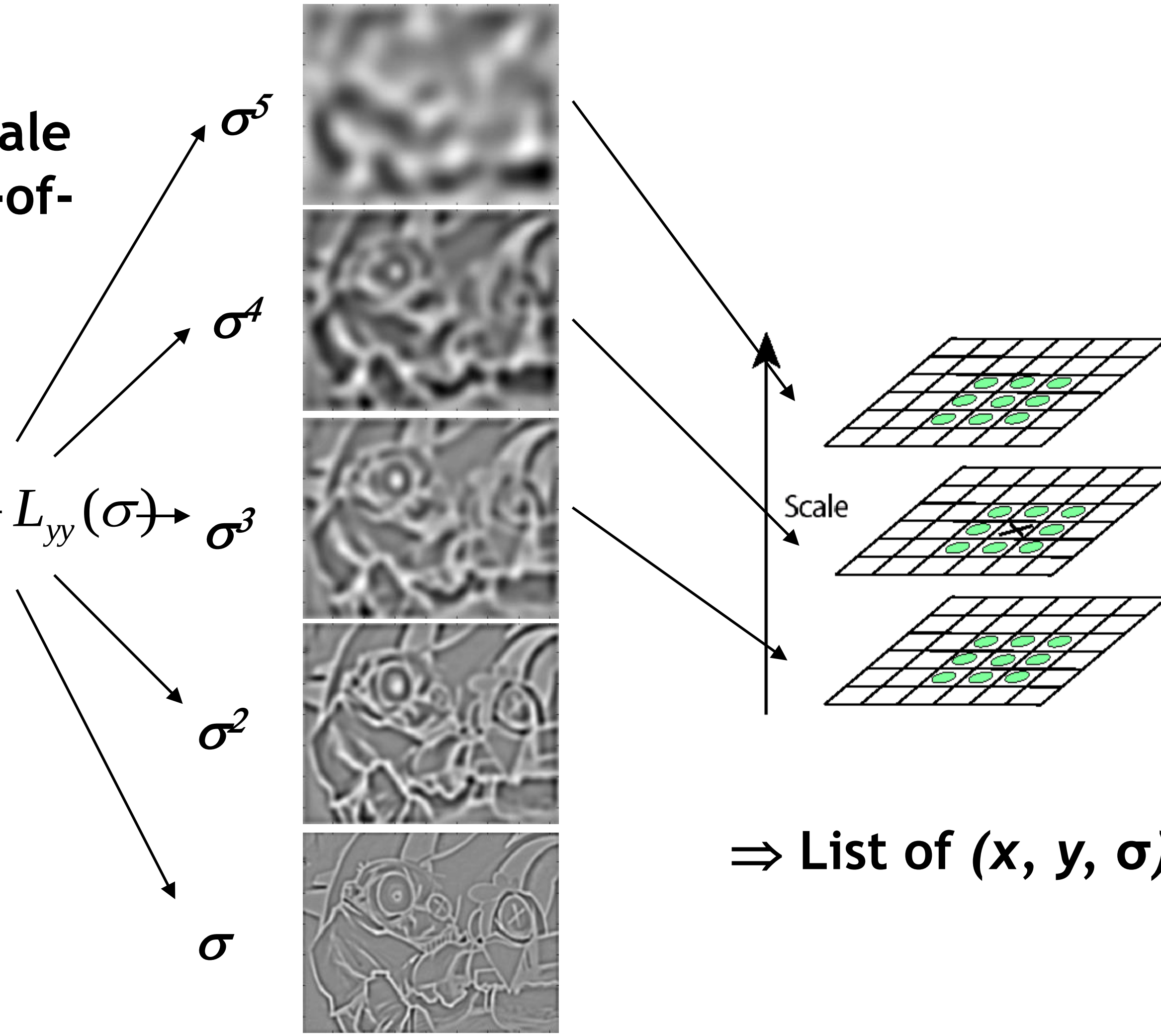
Slide credit: Krystian Mikolajczyk

Recap: Laplacian-of-Gaussian (LoG)

- **Interest points:**
  - Local maxima in scale space of Laplacian-of-Gaussian

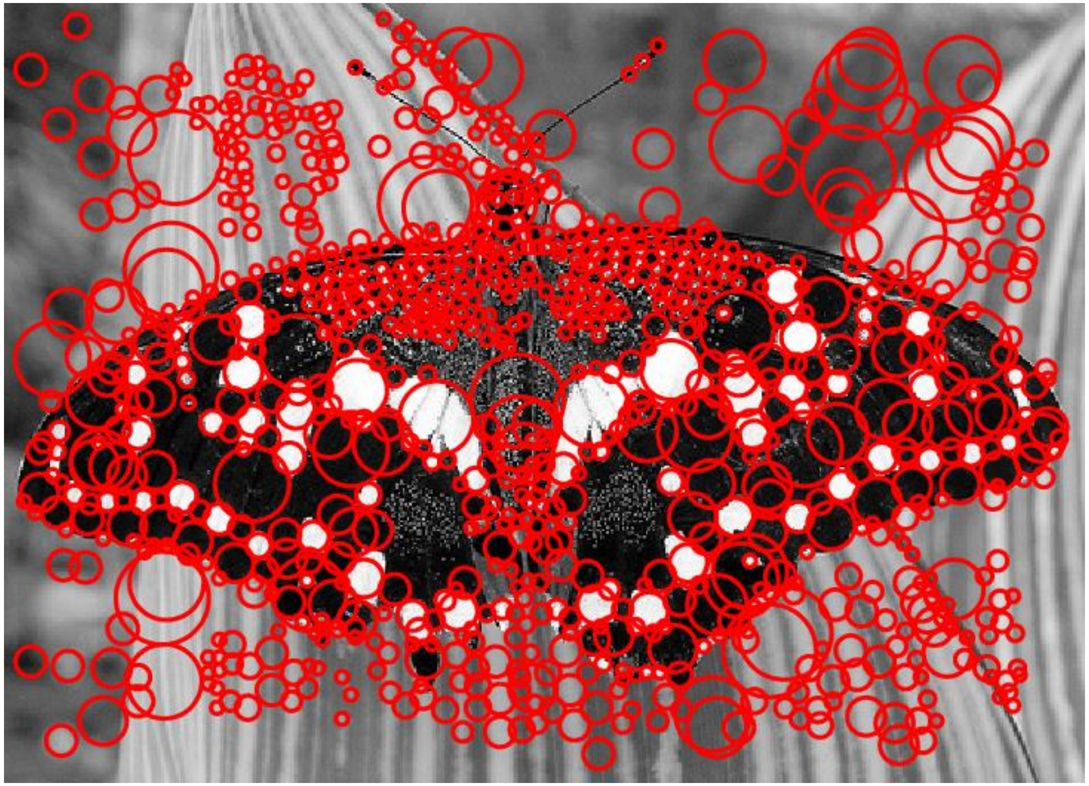


$$L_{xx}(\sigma) + L_{yy}(\sigma)$$



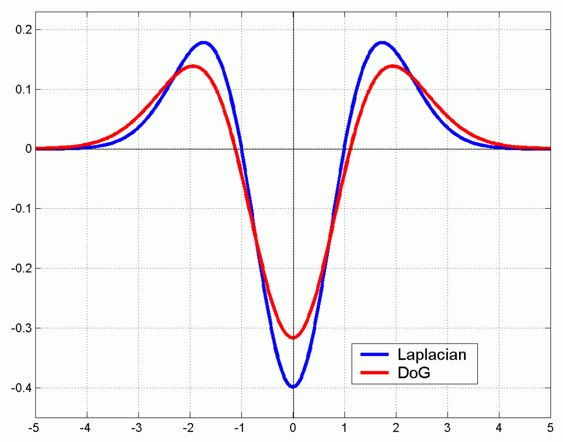
$\Rightarrow$ **List of *(x, y, σ)***

Recap: LoG Detector Responses

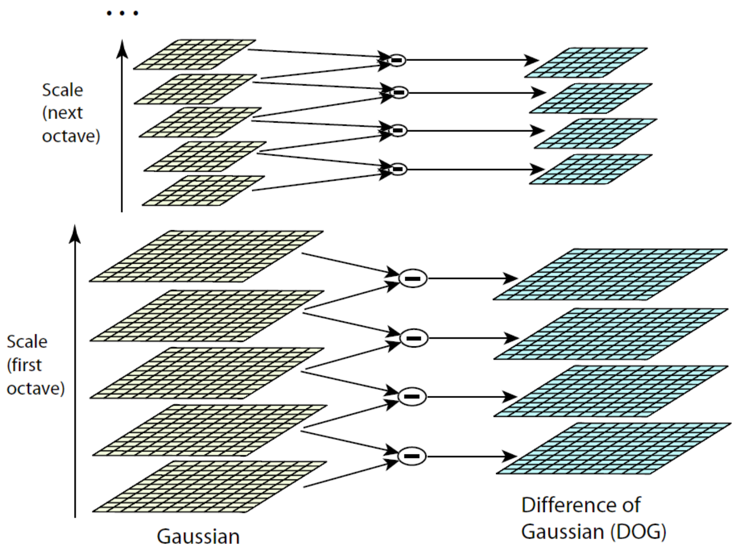


Recap: Key point localization with DoG

- **Efficient implementation**
  - Approximate LoG with a difference of Gaussians (DoG)



- **Approach DoG Detector**
  - Detect maxima of difference-of-Gaussian in scale space
  - Reject points with low contrast (threshold)
  - Eliminate edge responses



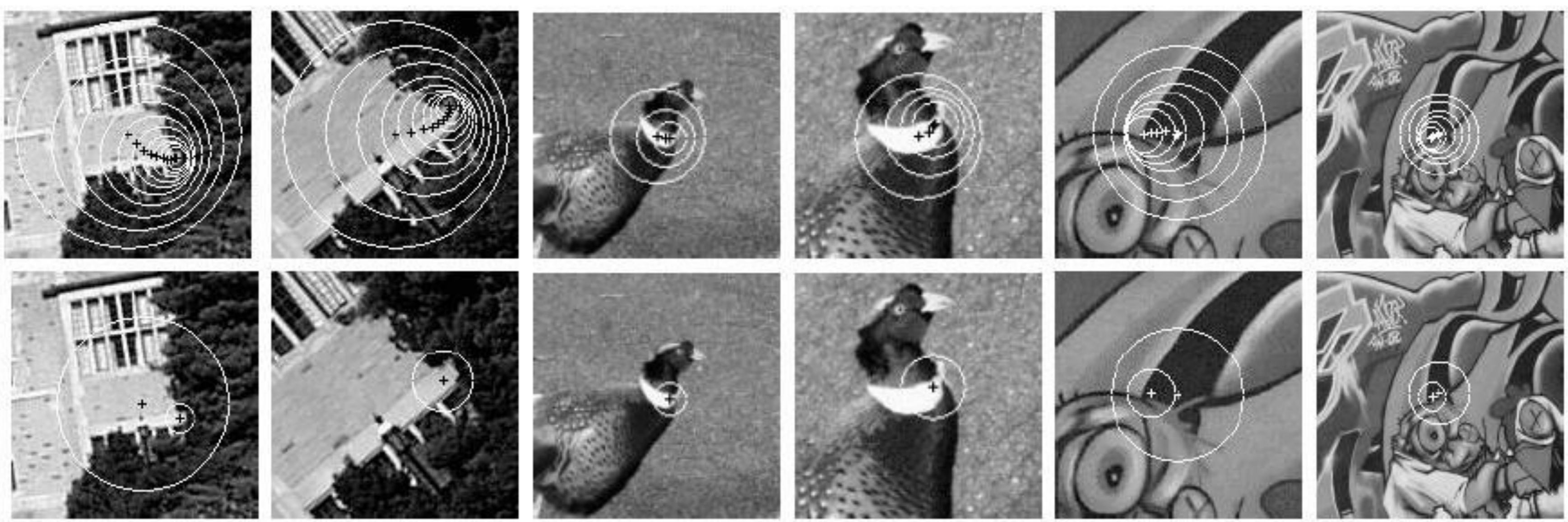
Candidate keypoints: list of (x,y,σ)

Image source: David Lowe

Recap: Harris-Laplace [Mikolajczyk '01]

1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian
   (same procedure with Hessian ⇒ Hessian-Laplace)

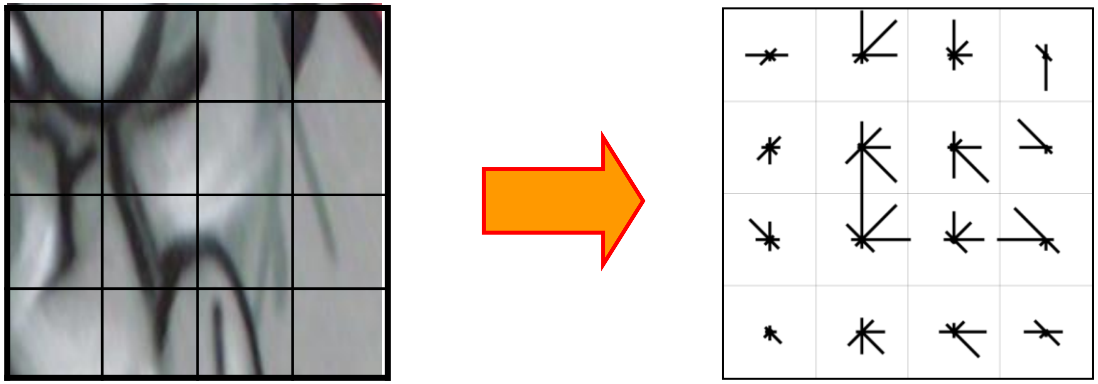
Harris points



Harris-Laplace points

Recap: SIFT Feature Descriptor

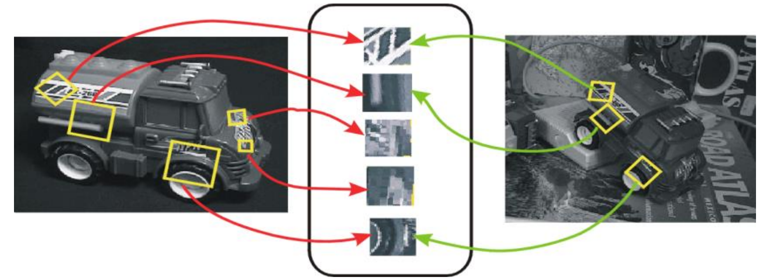
- **Scale Invariant Feature Transform**
- **Descriptor computation:**
  - Divide patch into 4x4 sub-patches: 16 cells
  - Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions



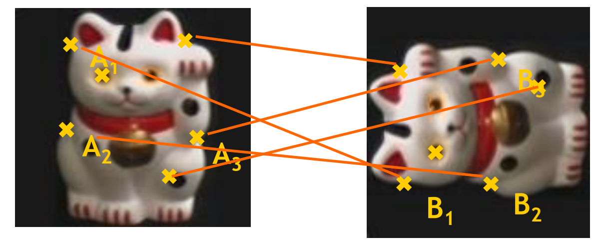
David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Repetition

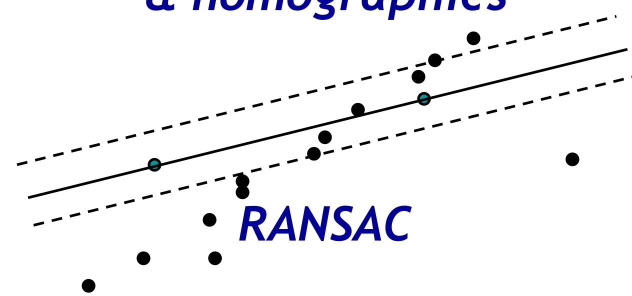
- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
  - Local Features - Detection and Description
  - Recognition with Local Features
- Object Categorization
- 3D Reconstruction
- Motion and Tracking



Recognition pipeline



Fitting affine transformations & homographies

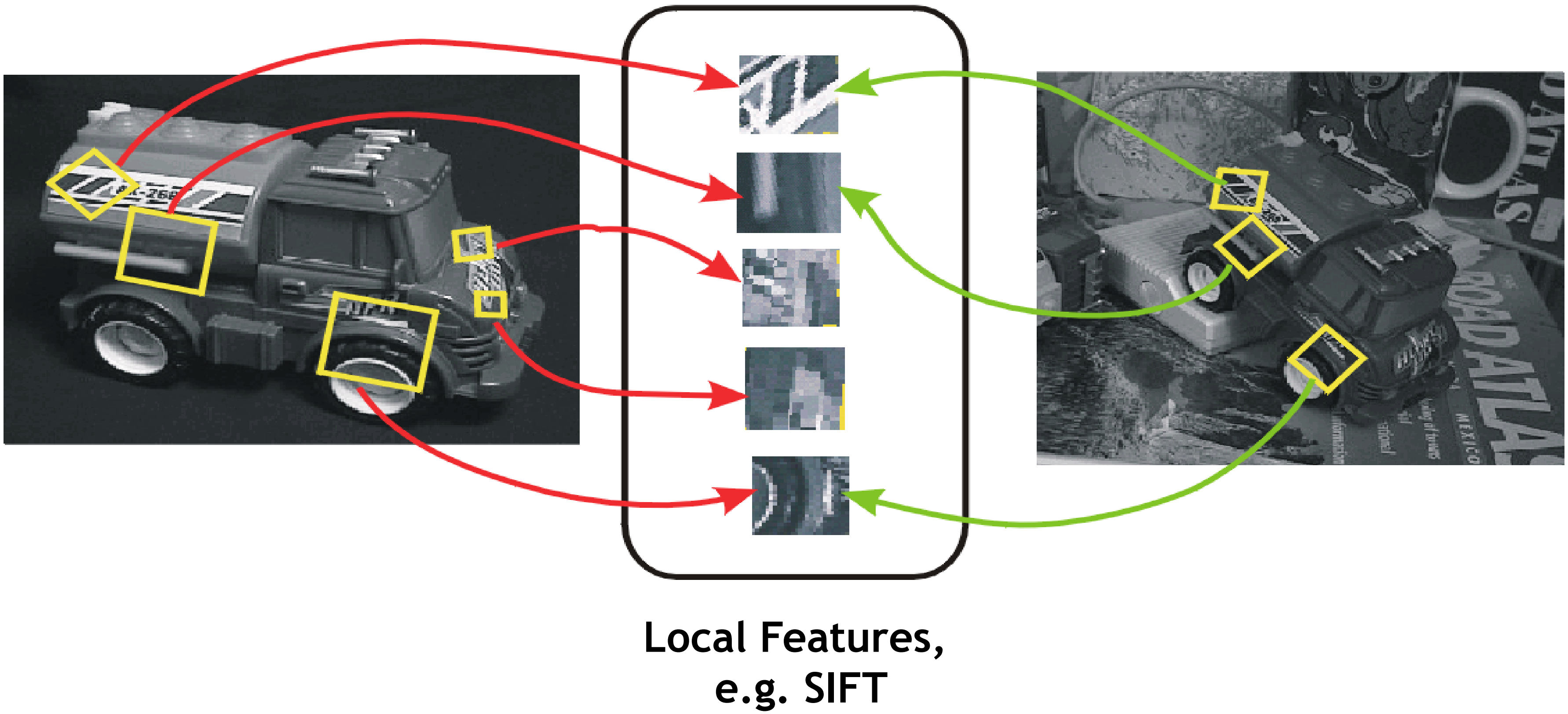


RANSAC

Gen. Hough Transform

Recap: Recognition with Local Features

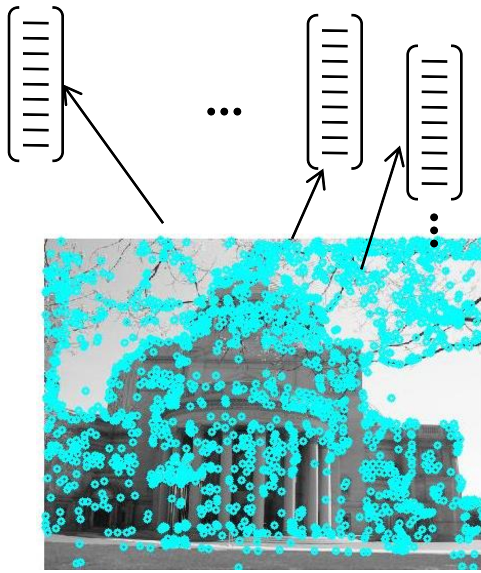
- Image content is transformed into local features that are invariant to translation, rotation, and scale
- Goal: Verify if they belong to a consistent configuration



Slide credit: David Lowe

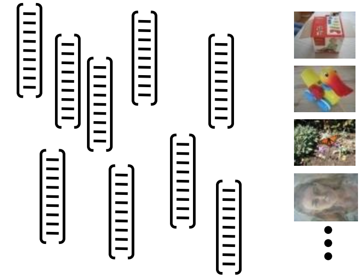
Recap: Indexing features

Detect or sample features



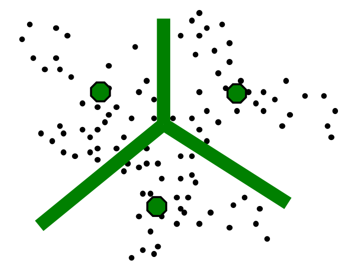
Describe features

List of positions, scales, orientations → Associated list of d-dimensional descriptors



Index each one into pool of descriptors from previously seen images

or

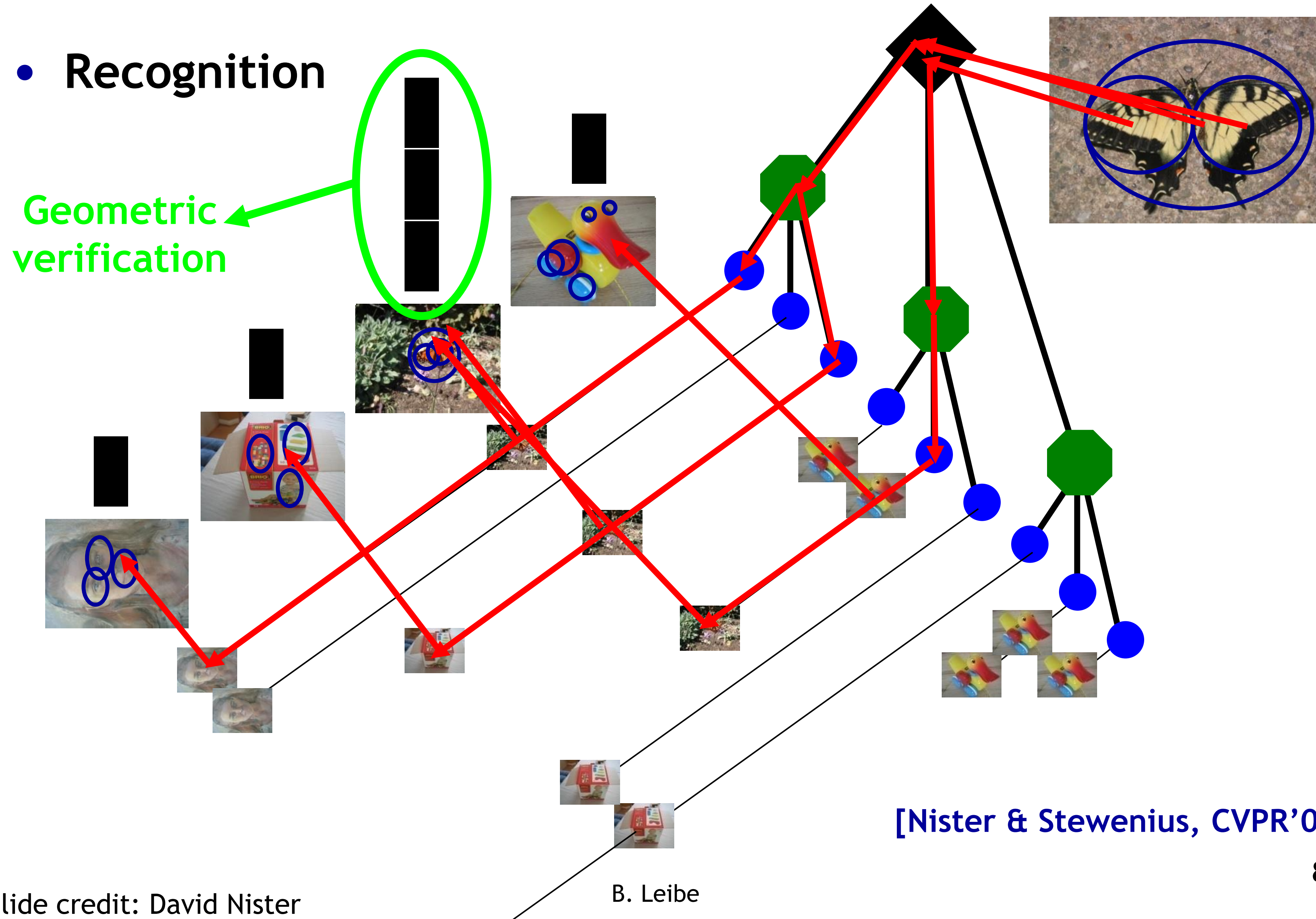


Match to quantized descriptors (visual words)

⇒ *Shortlist of possibly matching images + feature correspondences*

Recap: Fast Indexing with Vocabulary Trees

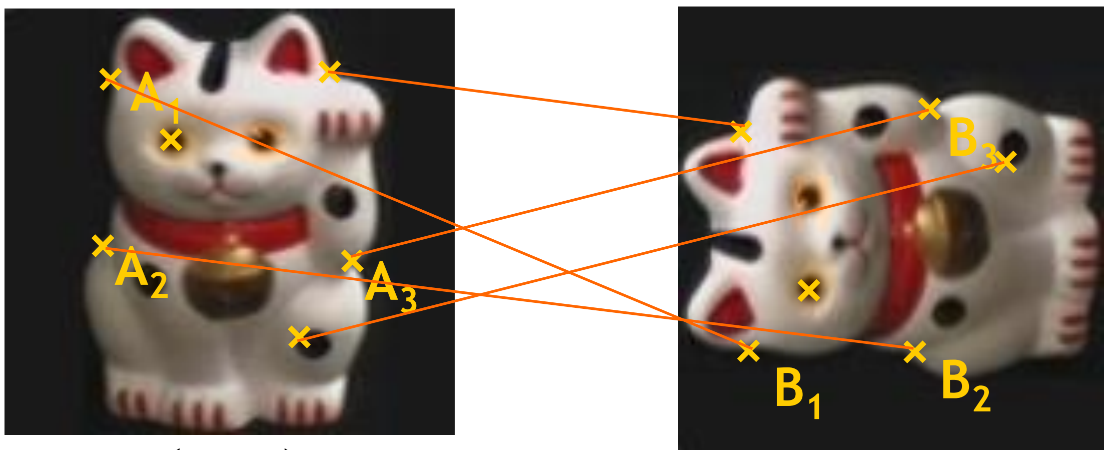
- **Recognition**



[Nister & Stewenius, CVPR'06]

Recap: Fitting an Affine Transformation

- Assuming we know the correspondences, how do we get the transformation?



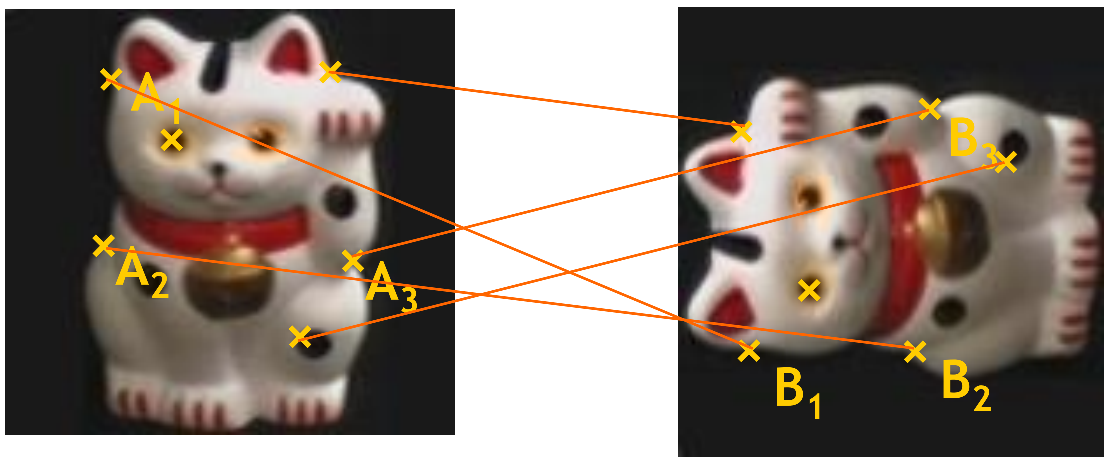
(x_i, y_i) $\qquad$ (x_i', y_i')

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

Recap: Fitting a Homography

- **Estimating the transformation**



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

Homogenous coordinates

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Image coordinates

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

Matrix notation

$$x' = Hx$$

$$x'' = \frac{1}{z'} x'$$

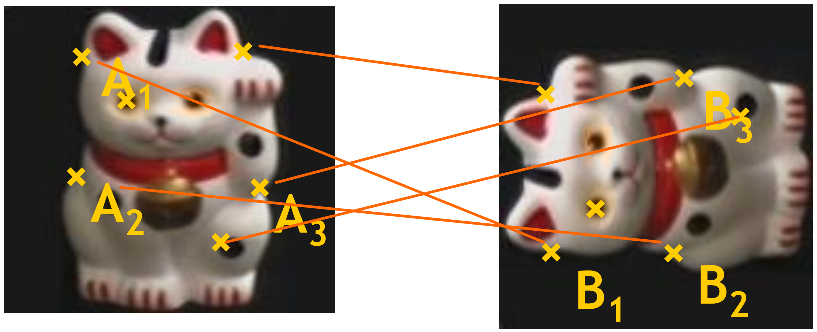
$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

$$y_{A_1} = \frac{h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

Recap: Fitting a Homography

see Exercise 5.5!

- **Estimating the transformation**



$$h_{11}\,x_{B_1} + h_{12} y_{B_1} + h_{13} - x_{A_1} h_{31}\,x_{B_1} - x_{A_1} h_{32} y_{B_1} - x_{A_1} = 0$$
$$h_{21}\,x_{B_1} + h_{22} y_{B_1} + h_{23} - y_{A_1} h_{31}\,x_{B_1} - y_{A_1} h_{32} y_{B_1} - y_{A_1} = 0$$

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

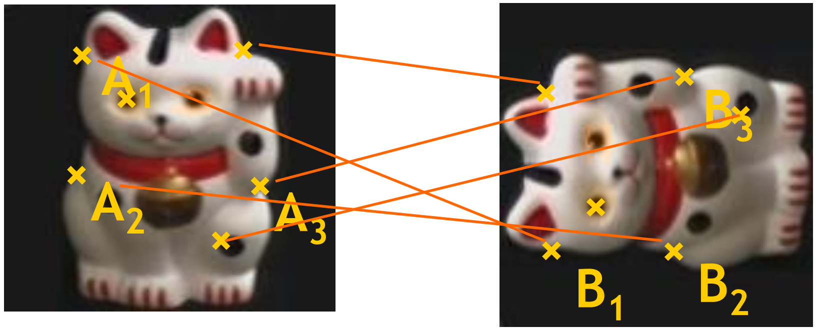
$$\begin{bmatrix} x_{B_1} & y_{B_1} & 1 & 0 & 0 & 0 & -x_{A_1}x_{B_1} & -x_{A_1}y_{B_1} & -x_{A_1} \\ 0 & 0 & 0 & x_{B_1} & y_{B_1} & 1 & -y_{A_1}x_{B_1} & -y_{A_1}y_{B_1} & -y_{A_1} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \end{bmatrix} \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ . \end{bmatrix}$$

$$Ah = 0$$

Slide credit: Krystian Mikolajczyk

Recap: Fitting a Homography

- Estimating the transformation
- Solution:
  - Null-space vector of A
  - Corresponds to smallest eigenvector



$$Ah = 0$$

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

SVD

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \mathbf{U}\begin{bmatrix} d_{11} & \cdots & d_{19} \\ \vdots & \ddots & \vdots \\ d_{91} & \cdots & d_{99} \end{bmatrix}\begin{bmatrix} v_{11} & \cdots & v_{19} \\ \vdots & \ddots & \vdots \\ v_{91} & \cdots & v_{99} \end{bmatrix}^T$$

$$\mathbf{h} = \frac{[v_{19}, \cdots, v_{99}]}{v_{99}}$$

Minimizes least square error

Slide credit: Krystian Mikolajczyk

Recap: RANSAC

see Exercise 6.2!

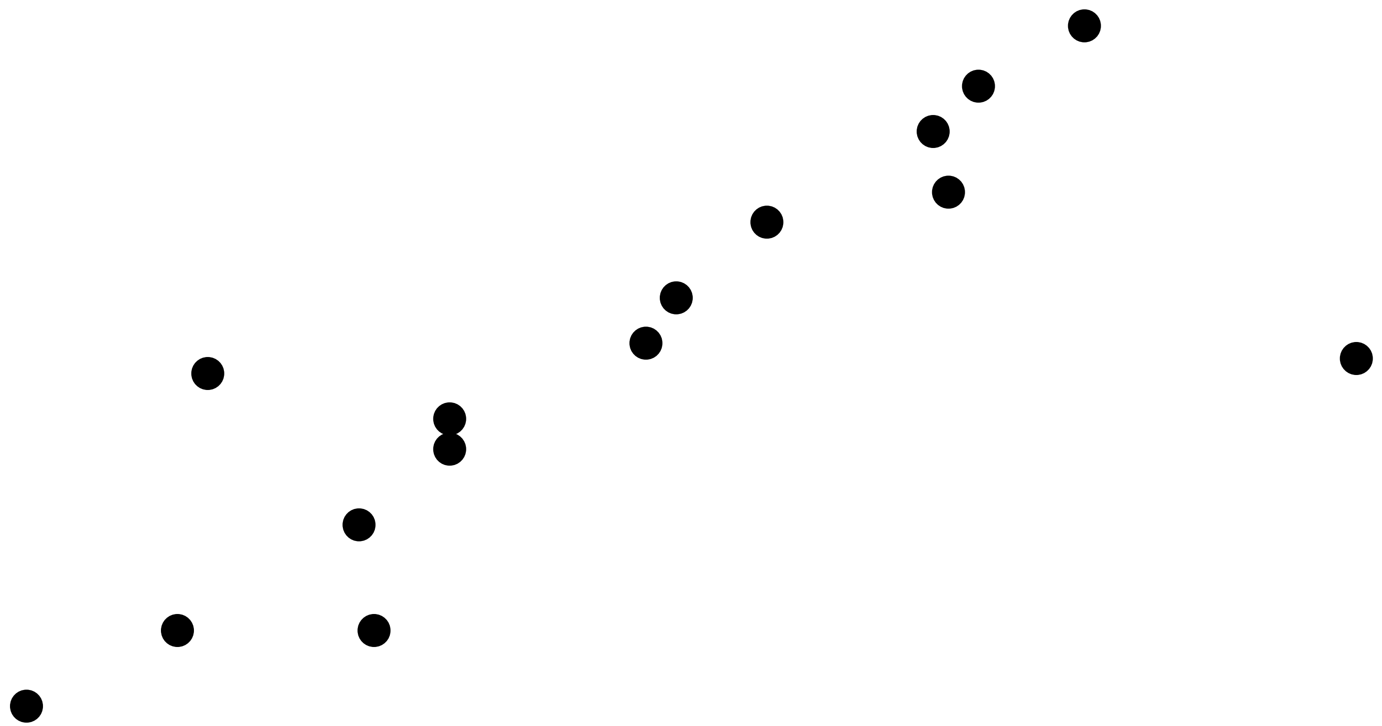
RANSAC loop:

1. Randomly select a *seed group* of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation from seed group
3. Find *inliers* to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers

- Keep the transformation with the largest number of inliers

Recap: RANSAC Line Fitting Example

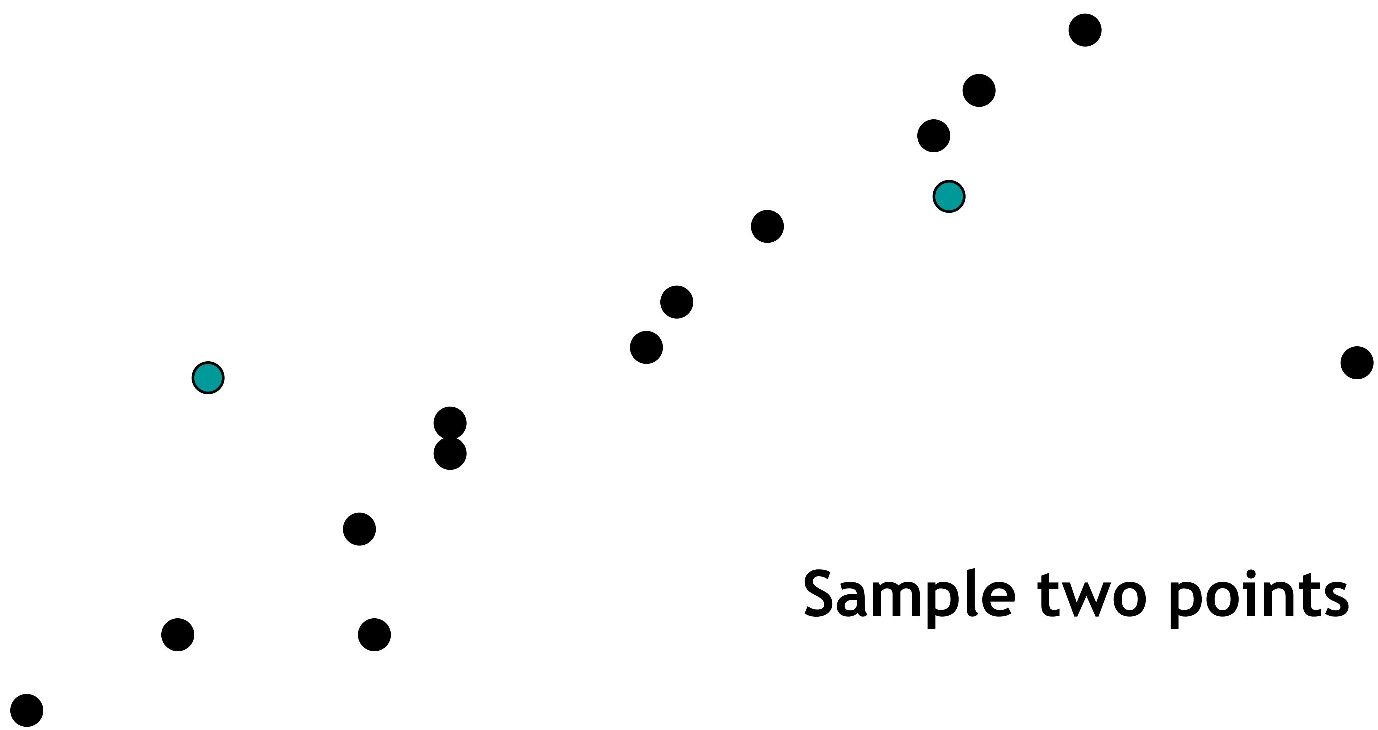
- **Task: Estimate the best line**



Slide credit: Jinxiang Chai

Recap: RANSAC Line Fitting Example

- Task: Estimate the best line

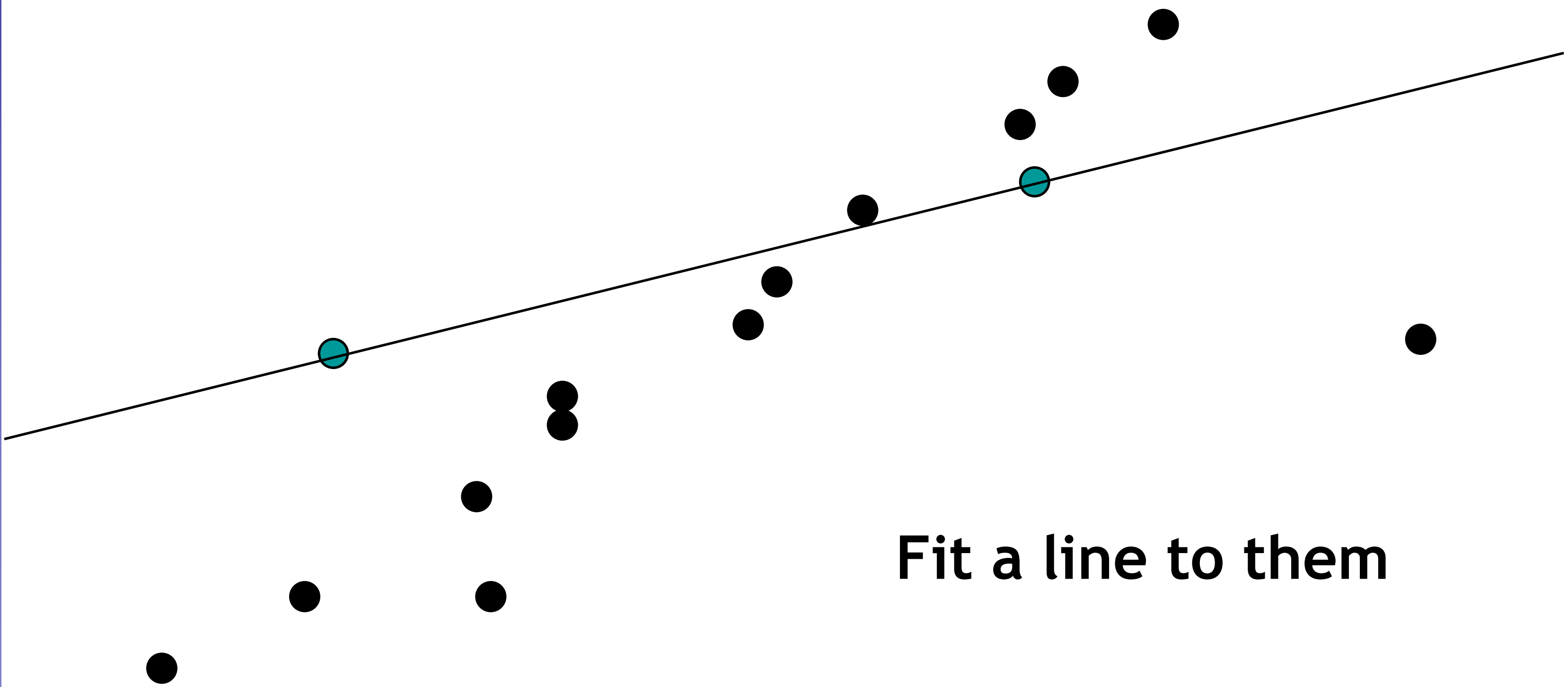


Sample two points

Slide credit: Jinxiang Chai

Recap: RANSAC Line Fitting Example

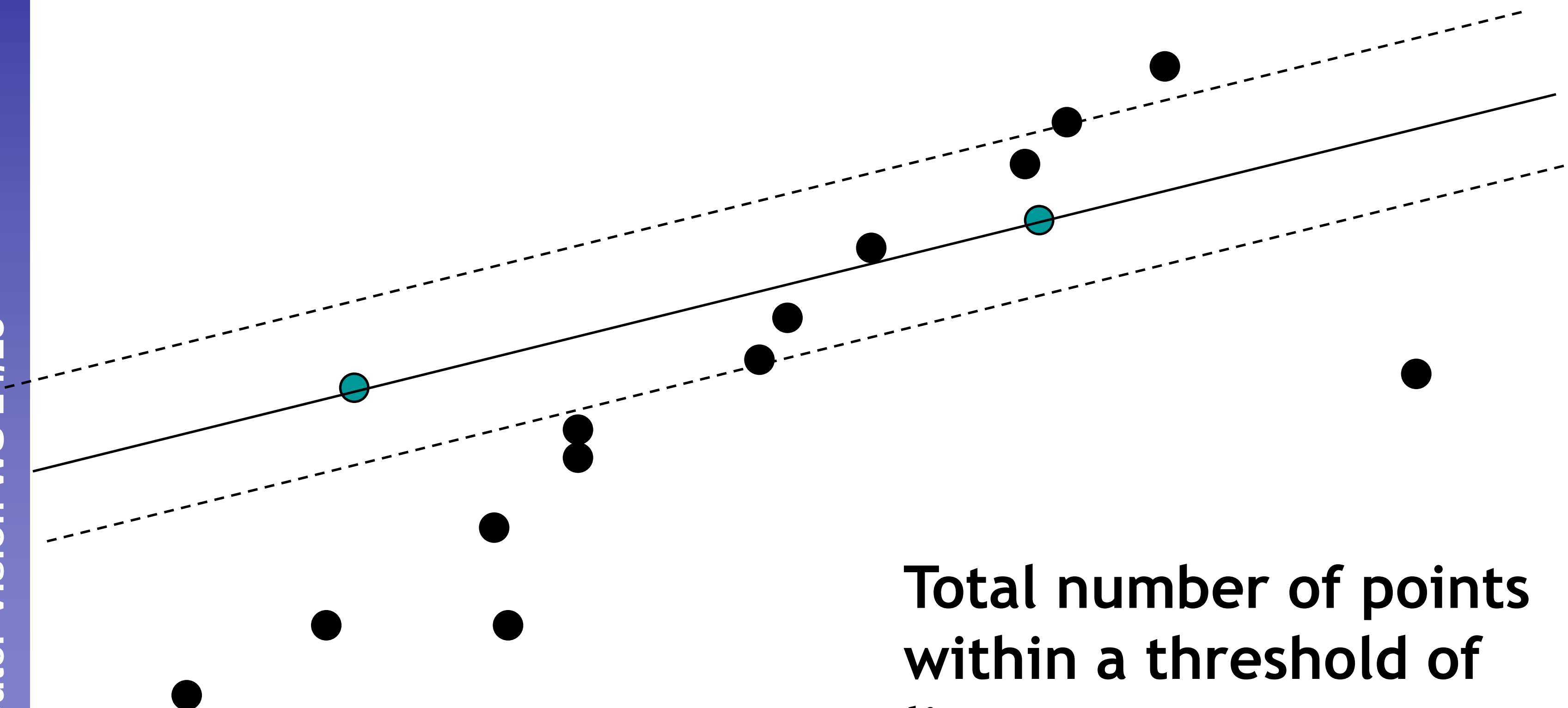
- **Task: Estimate the best line**



Fit a line to them

Recap: RANSAC Line Fitting Example

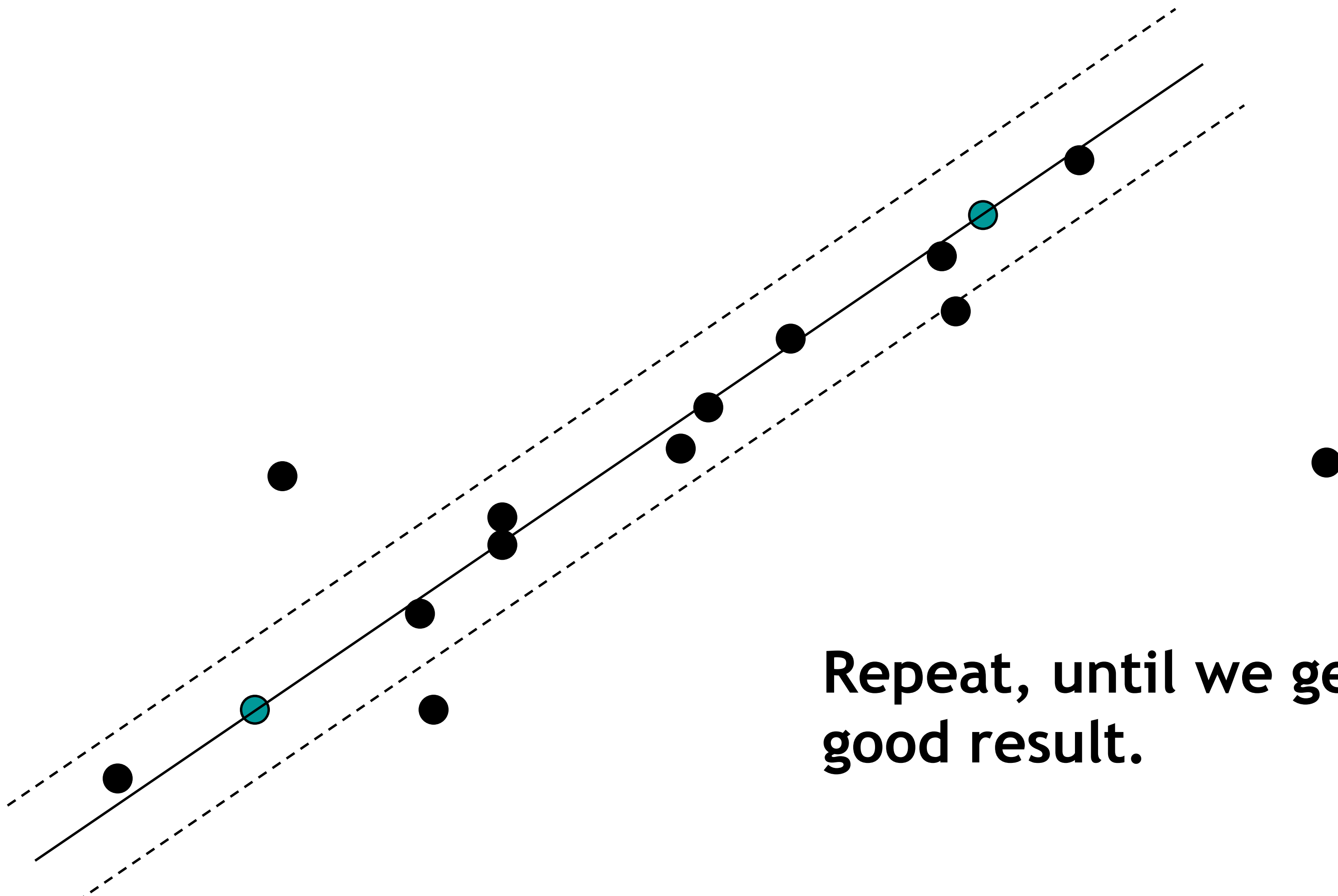
- Task: Estimate the best line



Total number of points within a threshold of line.

Recap: RANSAC Line Fitting Example

- Task: Estimate the best line



Repeat, until we get a good result.

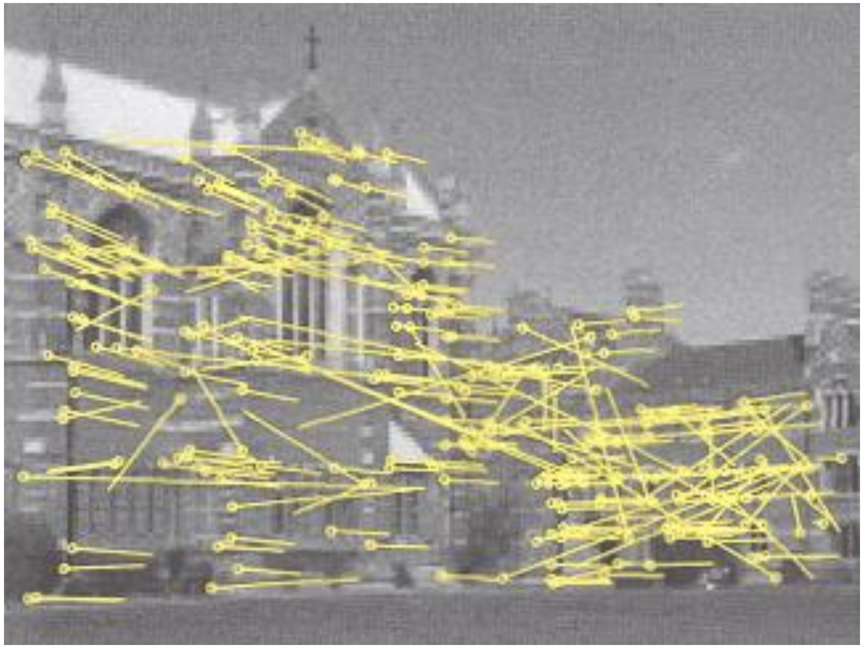
Slide credit: Jinxiang Chai

RWTH AACHEN UNIVERSITY

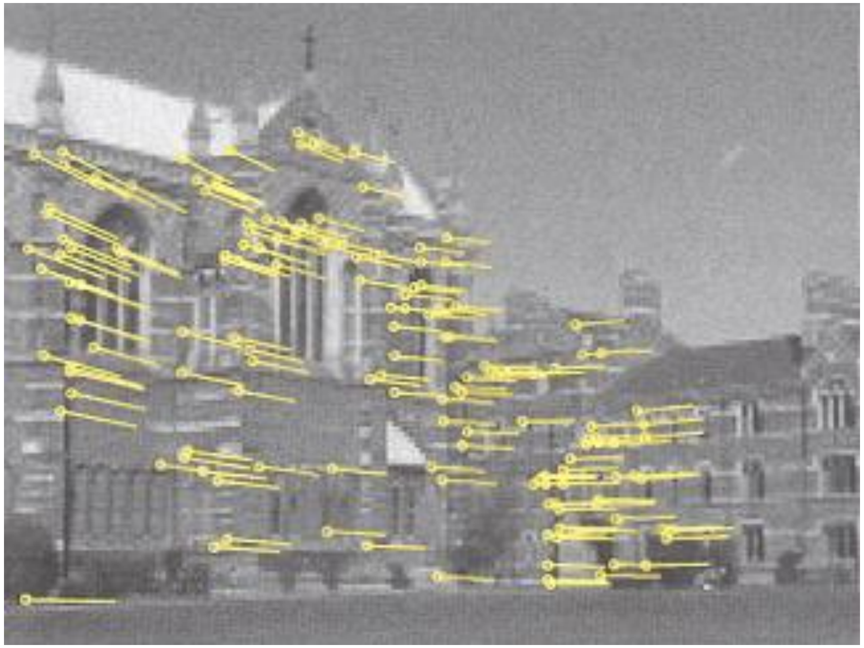
Recap: Feature Matching Example

- Find best stereo match within a square search window (here 300 pixels^2)
- Global transformation model: epipolar geometry

before RANSAC



after RANSAC



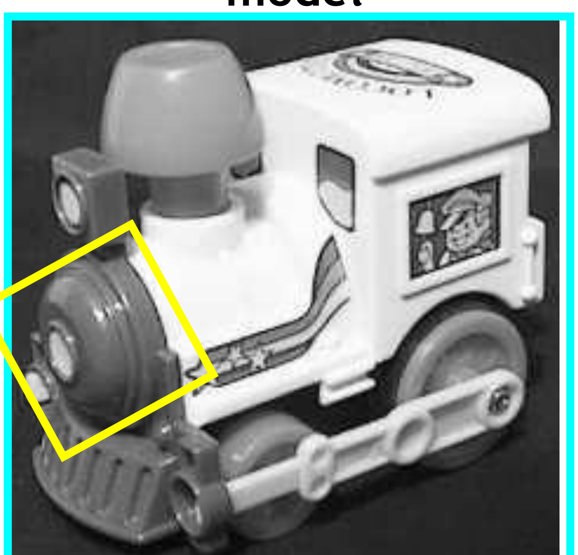
Images from Hartley & Zisserman

Slide credit: David Lowe

Recap: Generalized Hough Transform

- **Suppose our features are scale- and rotation-invariant**
  - **Then a single feature match provides an alignment hypothesis (translation, scale, orientation).**

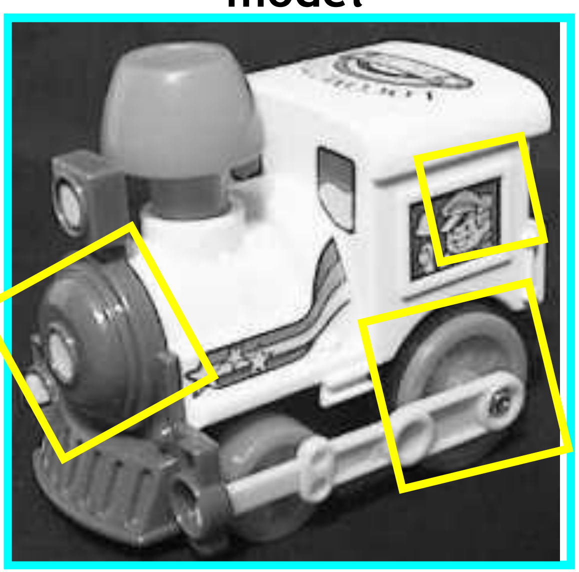
model



Recap: Generalized Hough Transform

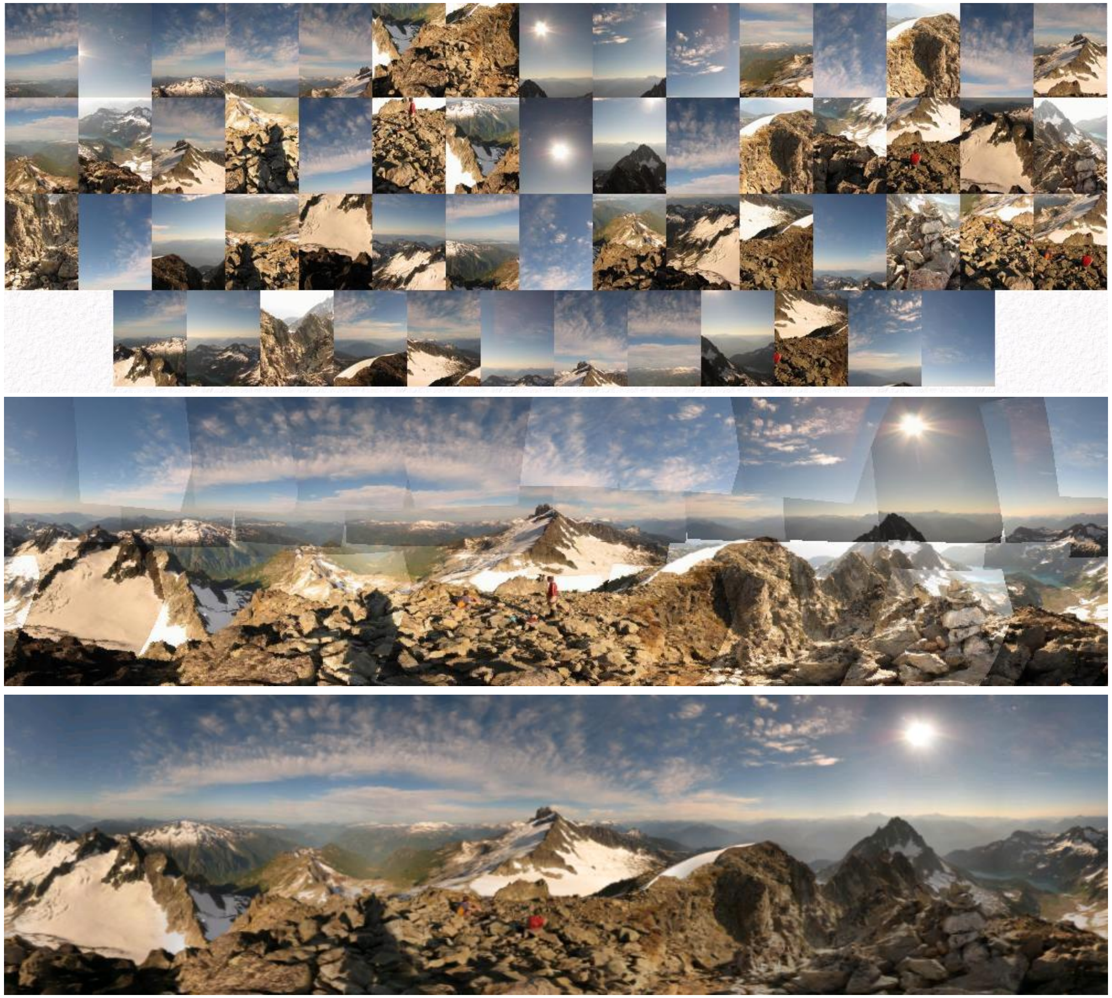
- Suppose our features are scale- and rotation-invariant
  - Then a single feature match provides an alignment hypothesis (translation, scale, orientation).
  - Of course, a hypothesis from a single match is unreliable.
  - Solution: let each match vote for its hypothesis in a Hough space with very coarse bins.

model



Application: Panorama Stitching

see Panorama Demo!

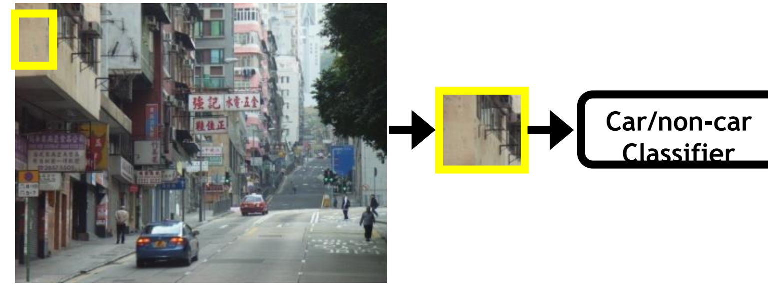


http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

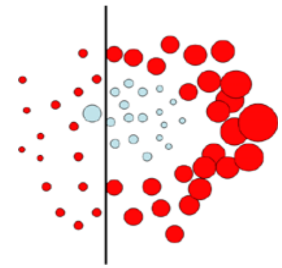
[Brown & Lowe, ICCV'03]

Repetition

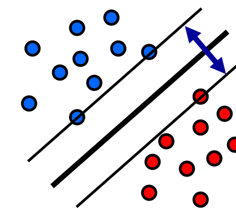
- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
  - **Sliding Window based Object Detection**
  - Bag-of-Words Approaches
- 3D Reconstruction
- Motion and Tracking



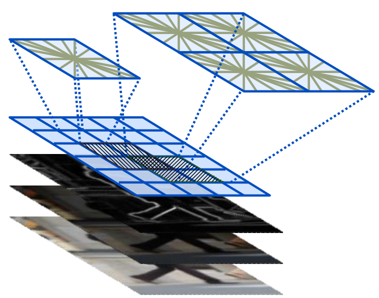
Sliding window principle



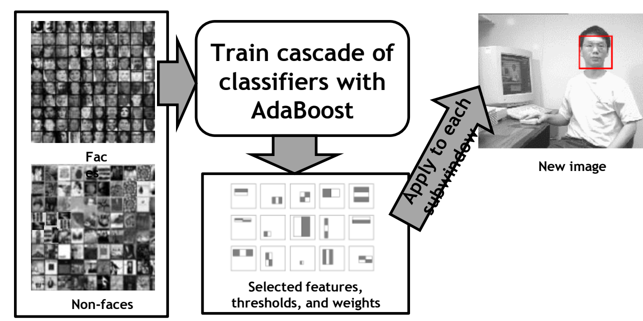
Boosting



SVM



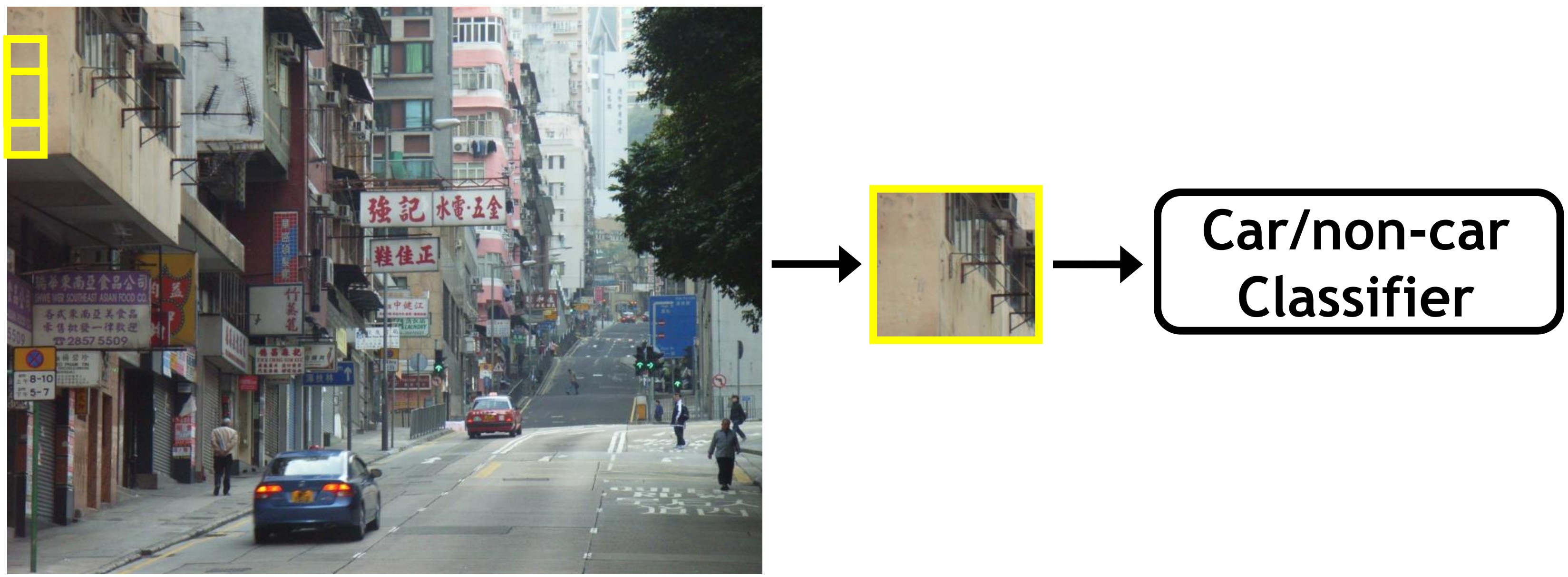
HOG detector



Viola-Jones face detector

Recap: Sliding-Window Object Detection

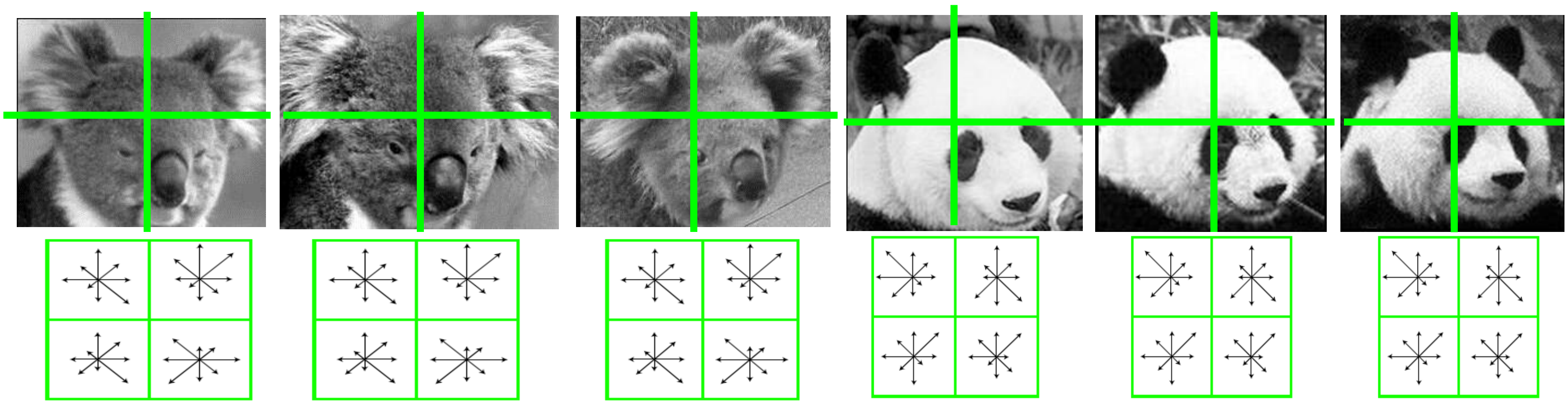
- If object may be in a cluttered scene, slide a window around looking for it.



- Essentially, this is a brute-force approach with many local decisions.

Recap: Gradient-based Representations

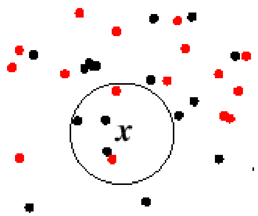
- Consider edges, contours, and (oriented) intensity gradients



- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Contrast-normalization: try to correct for variable illumination

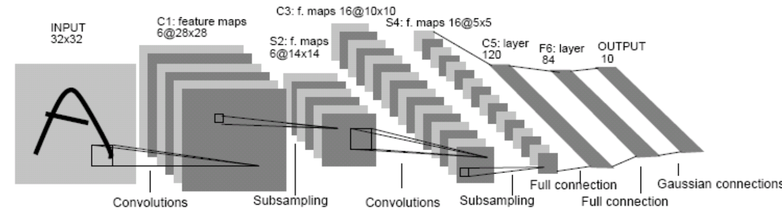
Recap: Classifier Construction: Many Choices…

Nearest Neighbor



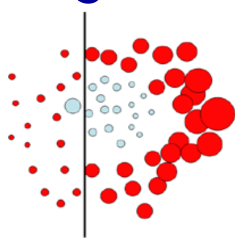
Berg, Berg, Malik 2005,
Chum, Zisserman 2007,
Boiman, Shechtman, Irani 2008, …

Neural networks



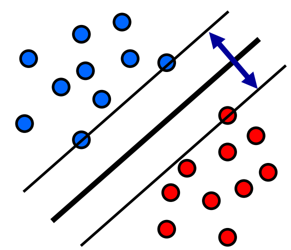
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Boosting



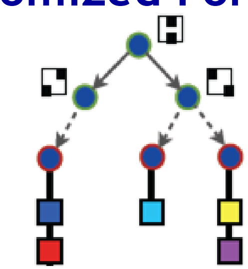
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,
Benenson 2012, …

Support Vector Machines



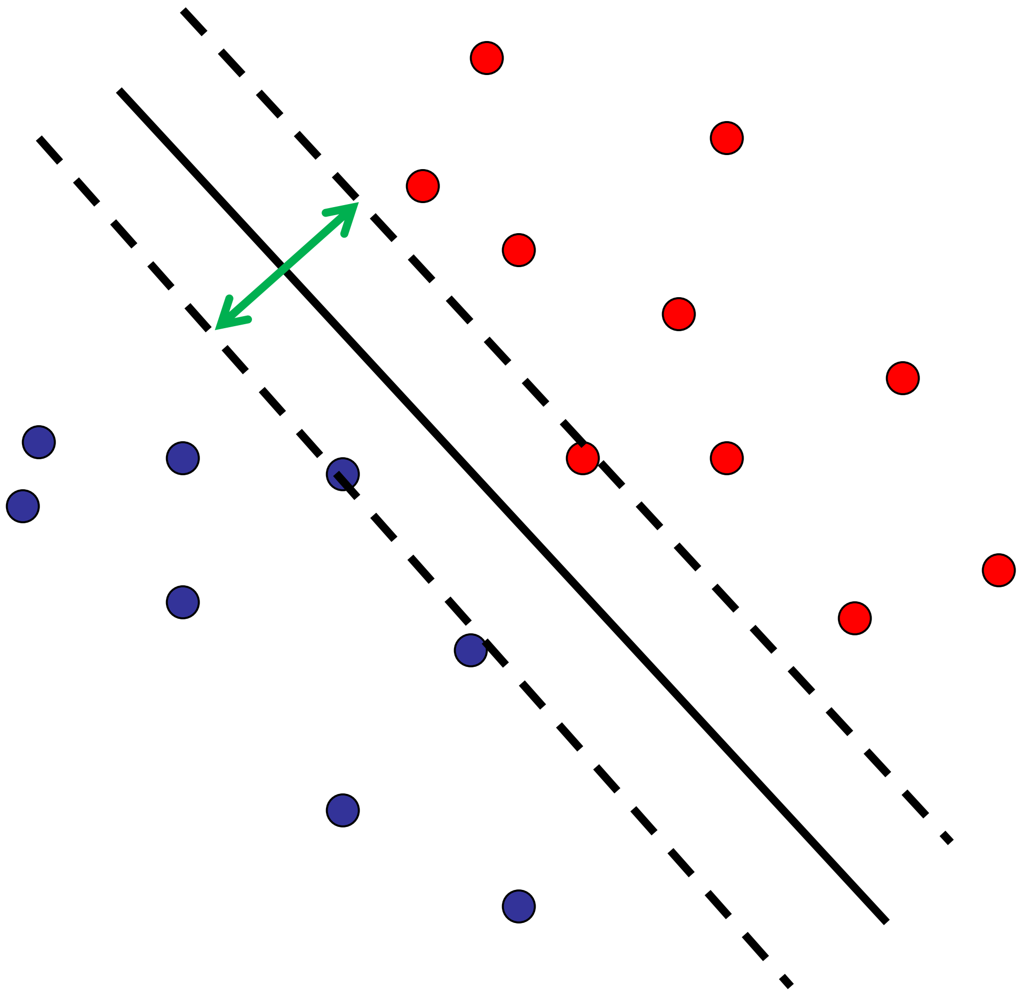
Vapnik, Schölkopf 1995,
Papageorgiou, Poggio '01,
Dalal, Triggs 2005,
Vedaldi, Zisserman 2012

Randomized Forests



Amit, Geman 1997,
Breiman 2001,
Lepetit, Fua 2006,
Gall, Lempitsky 2009,…

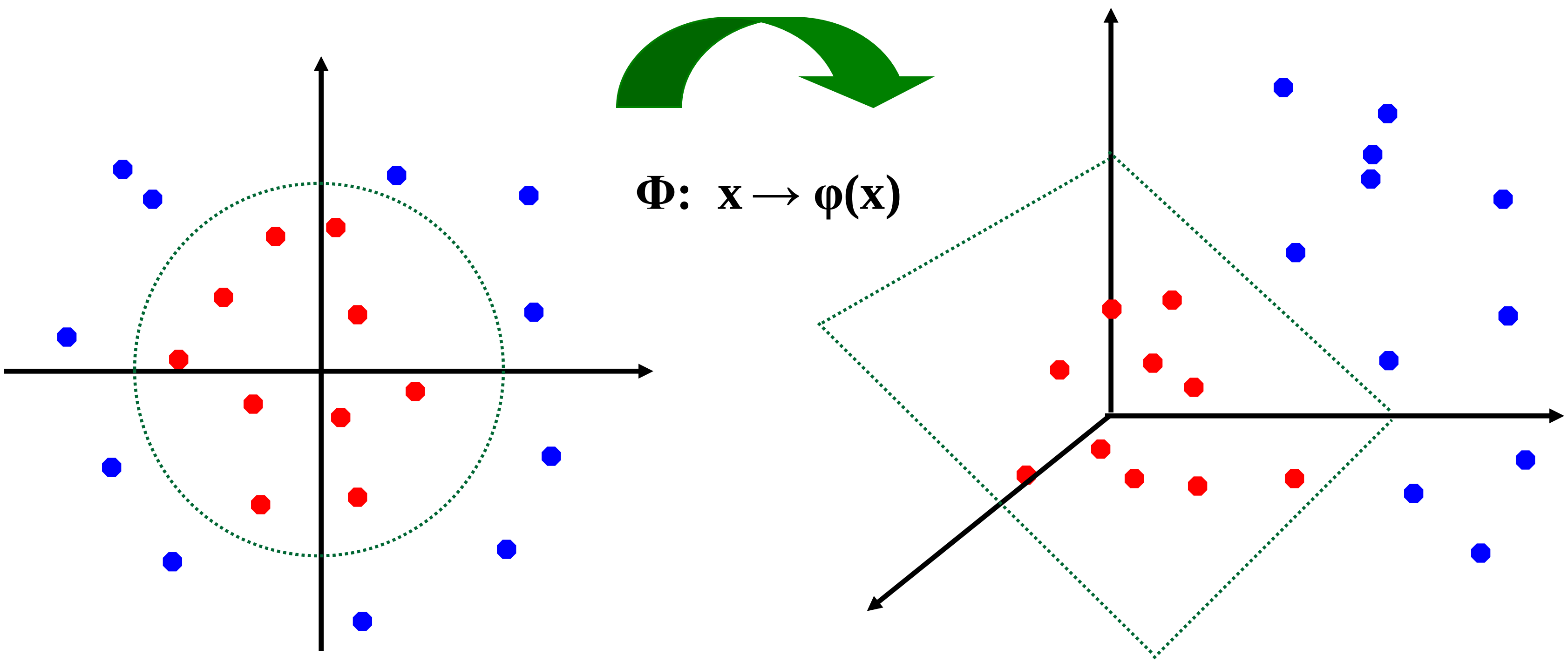
Recap: Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating hyperplane (i.e. line for 2D case)*
- Maximize the ***margin*** between the positive and negative training examples

Recap: Non-Linear SVMs

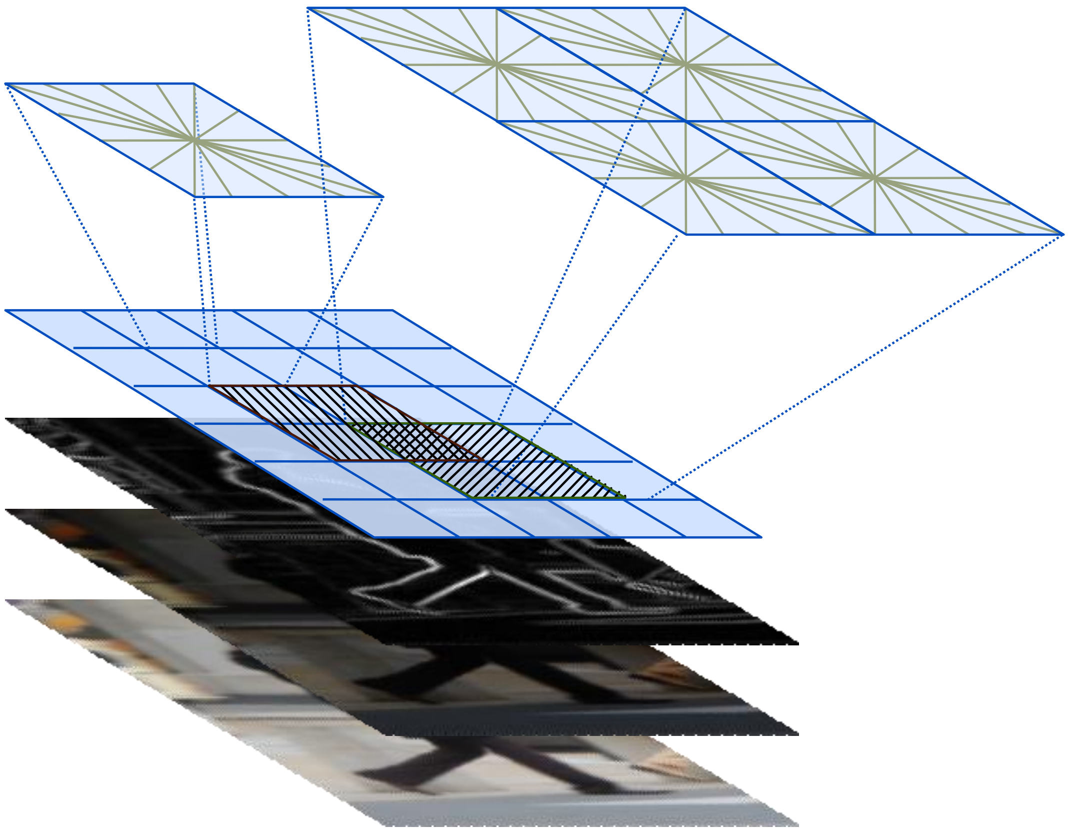
- General idea: The original input space can be mapped to some higher-dimensional feature space where the training set is separable:

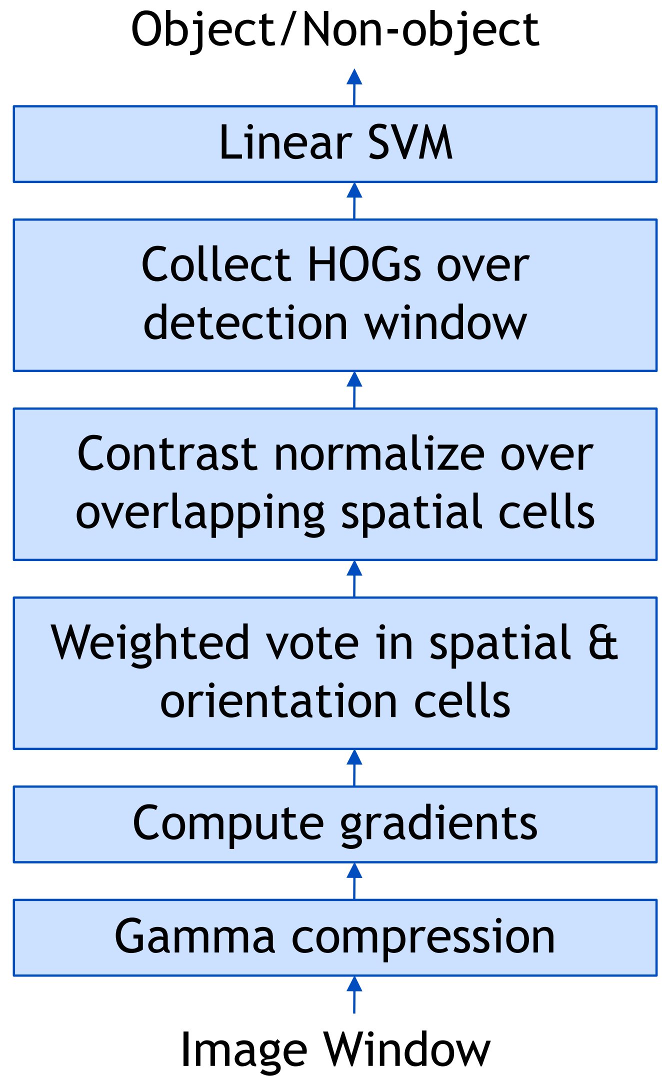


Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Recap: HOG Descriptor Processing Chain

- **SVM Classification**
  - Typically using a linear SVM





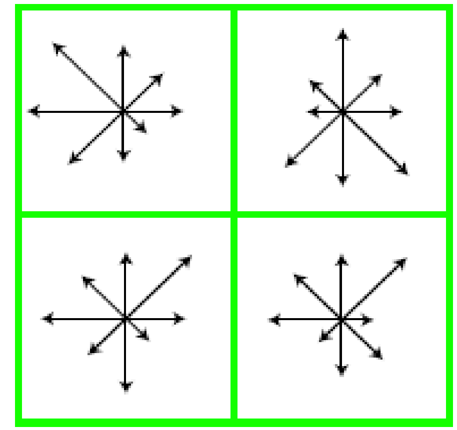
Slide adapted from Navneet Dalal

Recap: HOG Cell Computation Details

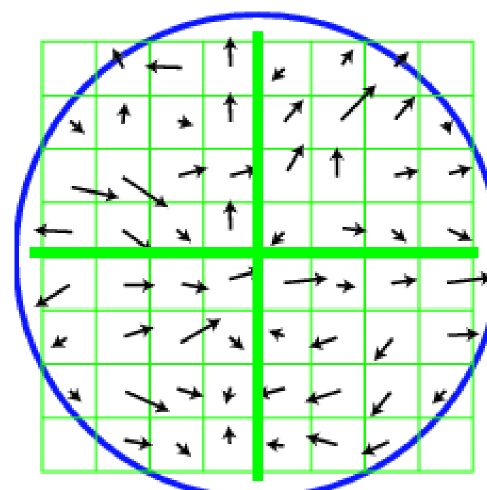
- **Gradient orientation voting**
  - Each pixel contributes to localized gradient orientation histogram(s)
  - Vote is weighted by the pixel's gradient magnitude

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$



- **Block-level Gaussian weighting**
  - An additional Gaussian weight is applied to each 2×2 block of cells
  - Each cell is part of 4 such blocks, resulting in 4 versions of the histogram.



Recap: HOG Cell Computation Details (2)

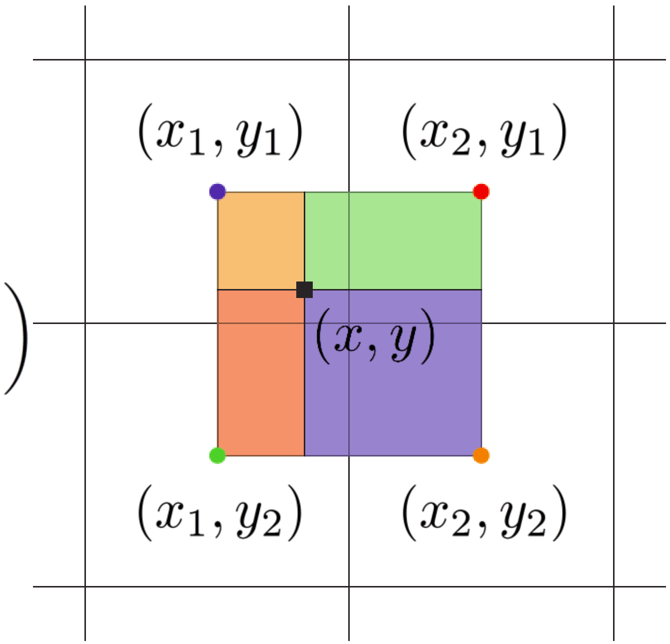
- Important for robustness: Tri-linear interpolation
  - Each pixel contributes to (up to) 4 neighboring cell histograms
  - Weights are obtained by bilinear interpolation in image space:

$$h(x_1, y_1) \leftarrow w \cdot \left(1 - \frac{x - x_1}{x_2 - x_1}\right)\left(1 - \frac{y - y_1}{y_2 - y_1}\right)$$

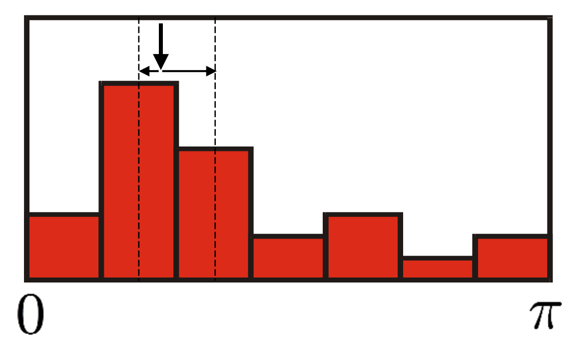
$$h(x_1, y_2) \leftarrow w \cdot \left(1 - \frac{x - x_1}{x_2 - x_1}\right)\left(\frac{y - y_1}{y_2 - y_1}\right)$$

$$h(x_2, y_1) \leftarrow w \cdot \left(\frac{x - x_1}{x_2 - x_1}\right)\left(1 - \frac{y - y_1}{y_2 - y_1}\right)$$

$$h(x_2, y_2) \leftarrow w \cdot \left(\frac{x - x_1}{x_2 - x_1}\right)\left(\frac{y - y_1}{y_2 - y_1}\right)$$



  - Contribution is further split over (up to) 2 neighboring orientation bins via linear interpolation over angles.



Recap: Non-Maximum Suppression

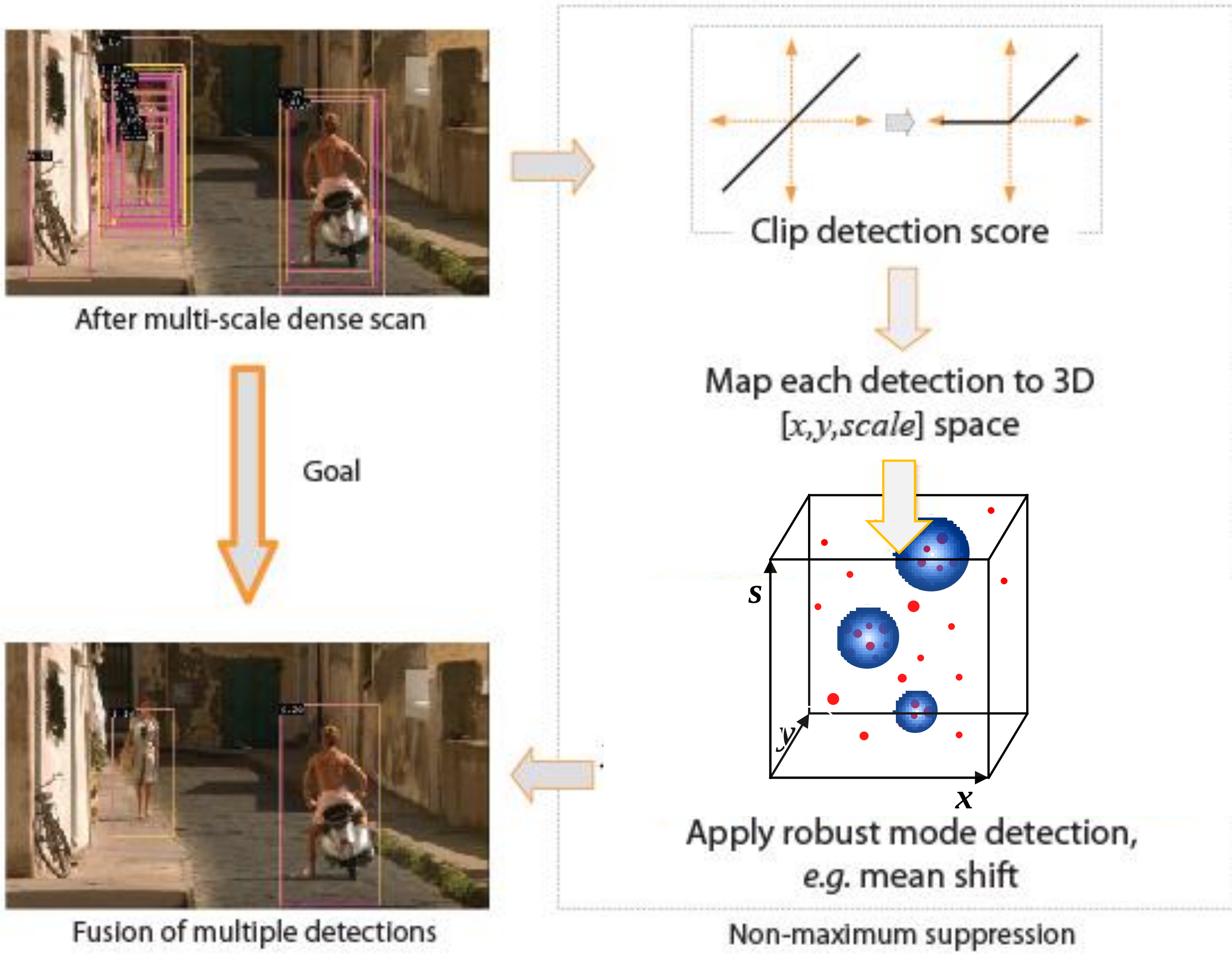
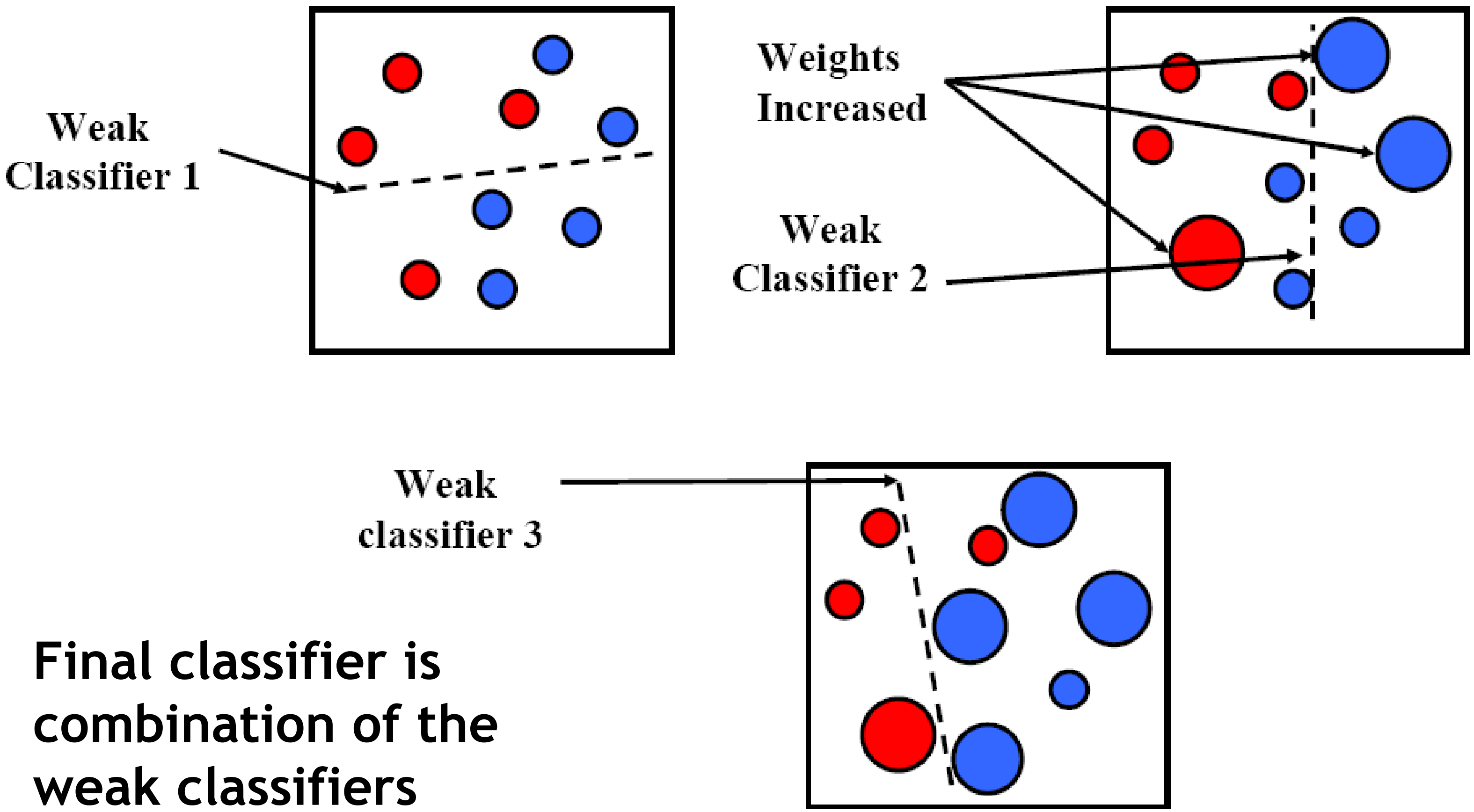


Image source: Navneet Dalal, PhD Thesis

RWTH AACHEN UNIVERSITY

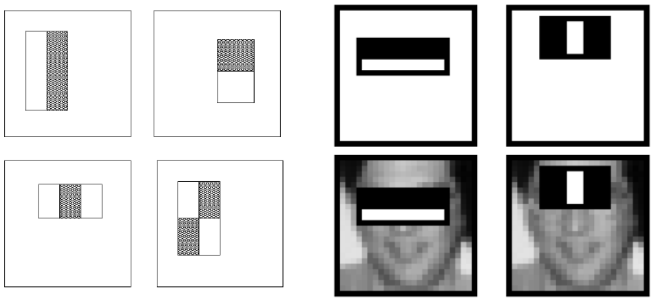
Recap: AdaBoost



Final classifier is combination of the weak classifiers

Recap: Viola-Jones Face Detection

"Rectangular" filters

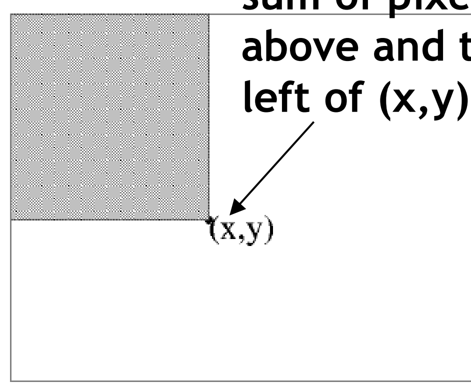


Feature output is difference between adjacent regions

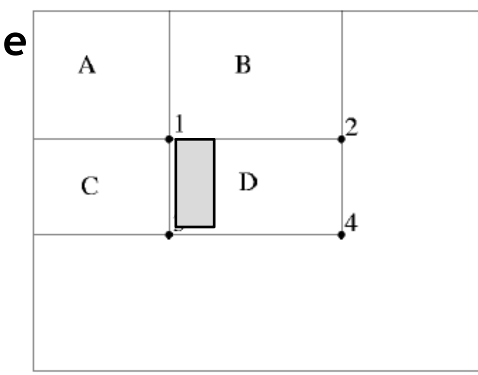
Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images → scale features directly for same cost

Value at (x,y) is sum of pixels above and to the left of (x,y)



Integral image

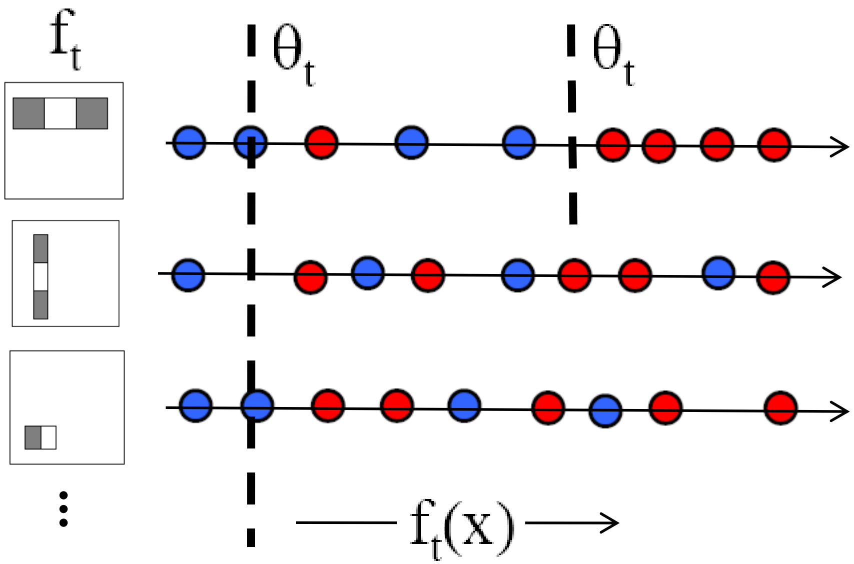


$$
\begin{aligned}
D &= 1 + 4 - (2 + 3) \\
&= A + (A + B + C + D) - (A + C + A + B) \\
&= D
\end{aligned}
$$

[Viola & Jones, CVPR 2001]

Recap: AdaBoost Feature+Classifier Selection

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of *weighted* error.



Outputs of a possible rectangle feature on faces and non-faces.

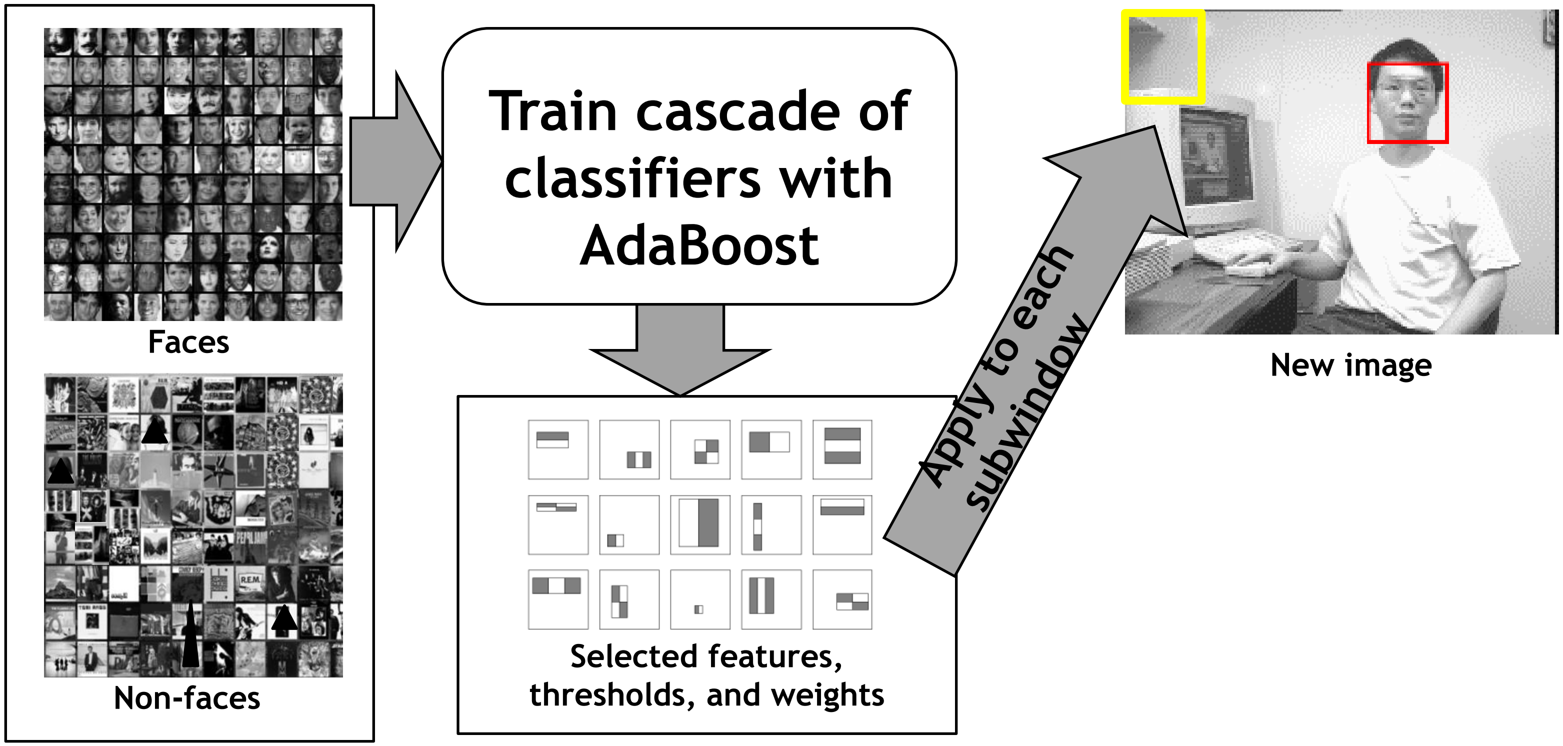
Resulting weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

[Viola & Jones, CVPR 2001]

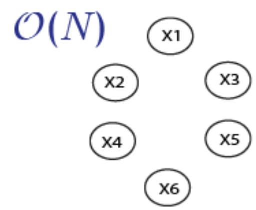
Application: Viola-Jones Face Detector

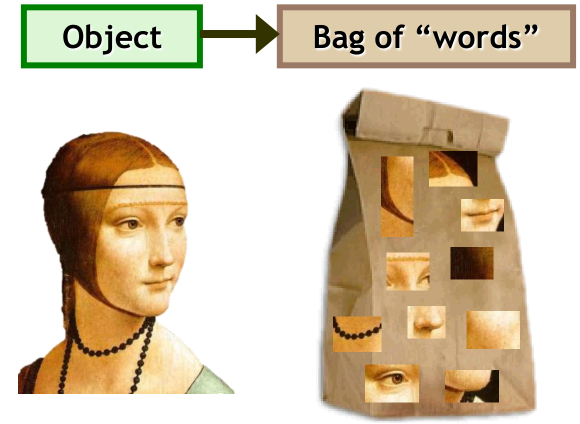


- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- [Implementation available in OpenCV: http://sourceforge.net/projects/opencvlibrary/]

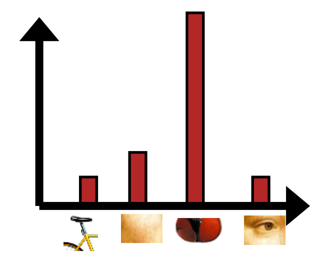
Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
  - Sliding Window based Object Detection
  - Part-based Approaches
- 3D Reconstruction
- Motion and Tracking

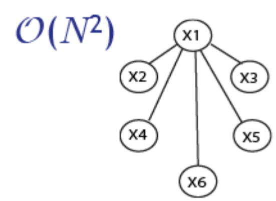


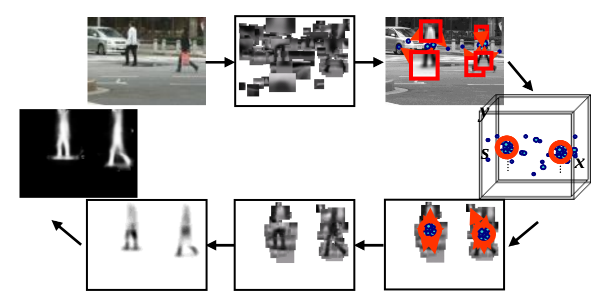


Bag-of-words representation



Activation histogram

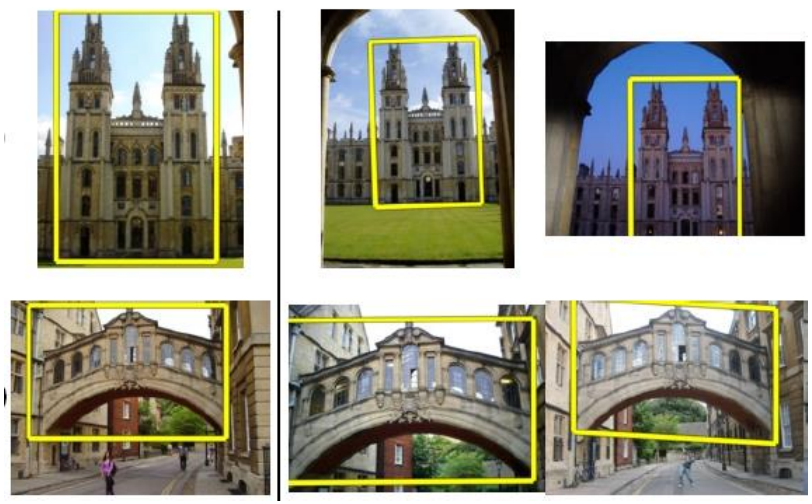




Implicit Shape Model

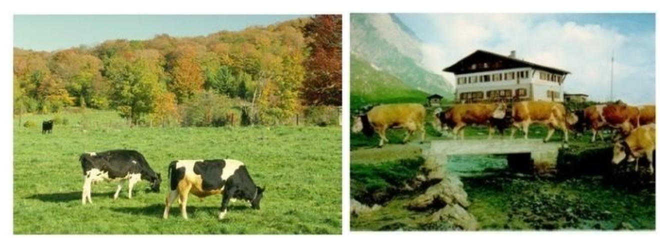
Recap: Identification vs. Categorization

- Find *this particular* object



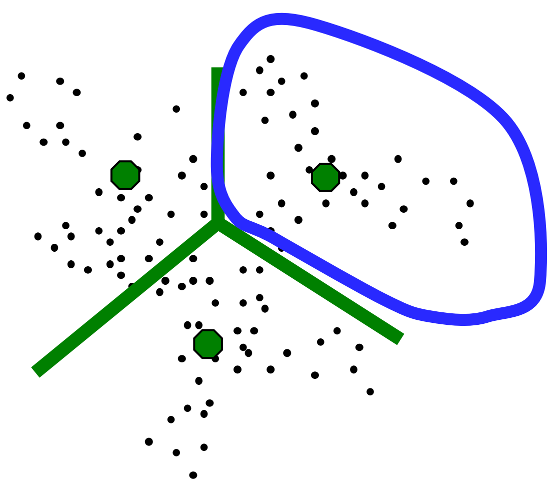
- Recognize ANY car

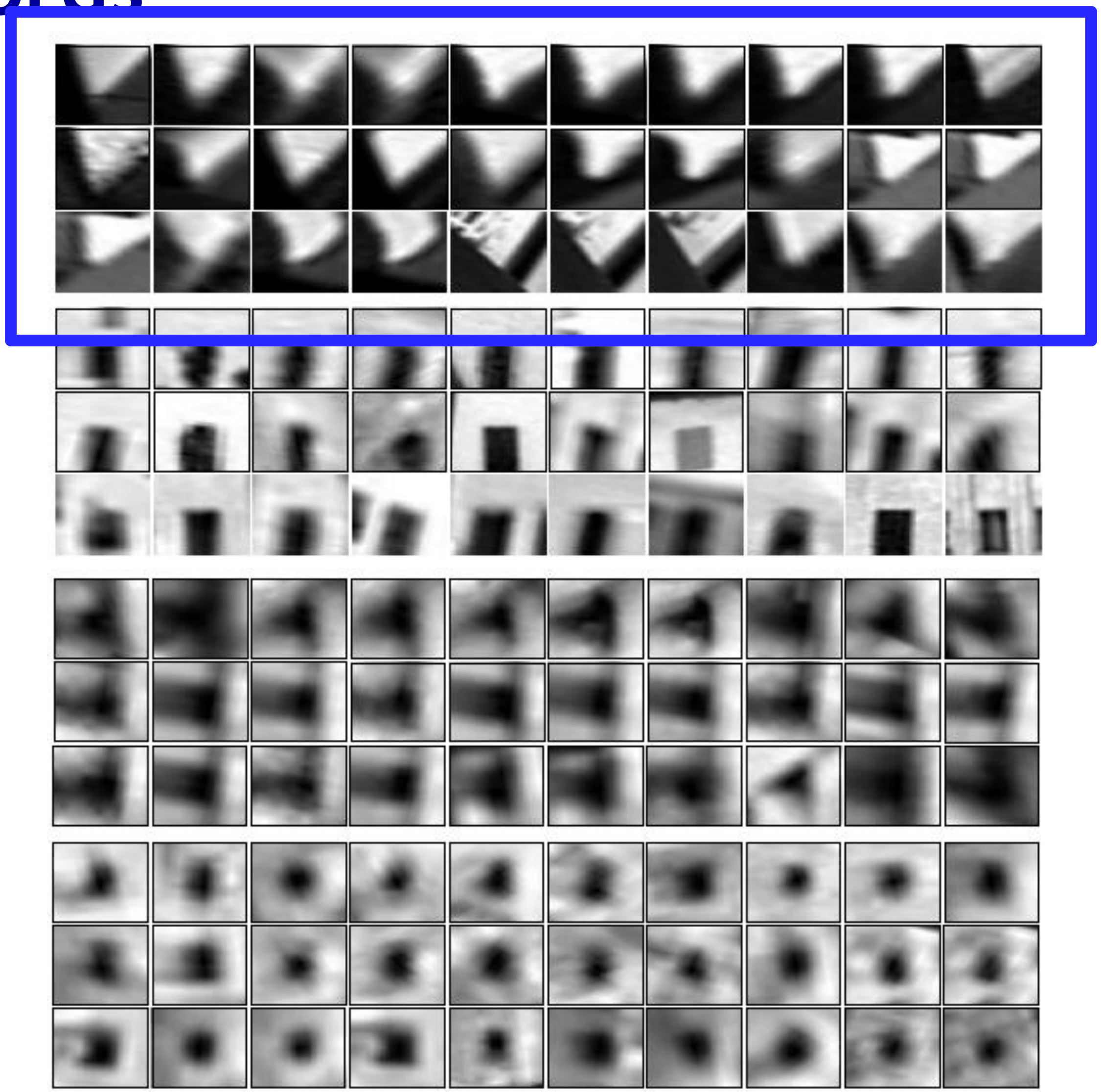
- Recognize ANY cow



Recap: Visual Words

- Quantize the feature space into “visual words”
- Perform matching only to those visual words.





Exact feature matching → Match to same visual word

Slide adapted from Kristen Grauman

Figure from Sivic & Zisserman, ICCV 2003

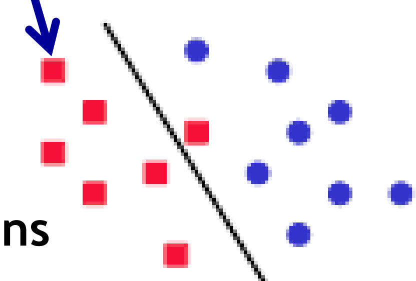
Recap: Bag-of-Word Representations (BoW)

Object → **Bag of “words”**

Source: ICCV 2005 short course, Li Fei-Fei

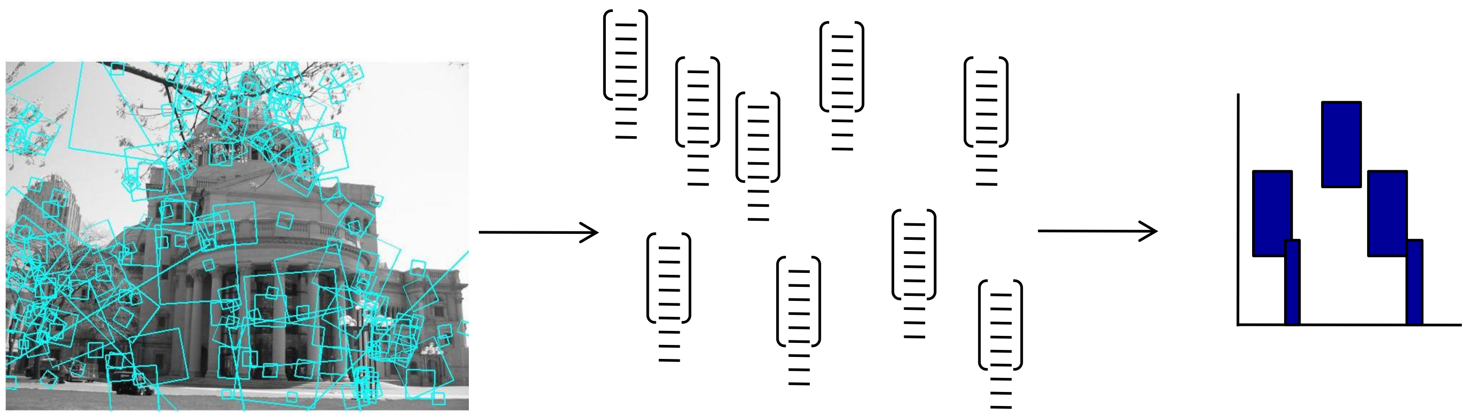
Recap: Categorization with Bags-of-Words

- Compute the word activation histogram for each image.
- Let each such BoW histogram be a feature vector.
- Use images from each class to train a classifier (e.g., an SVM).



Recap: Advantage of BoW Histograms

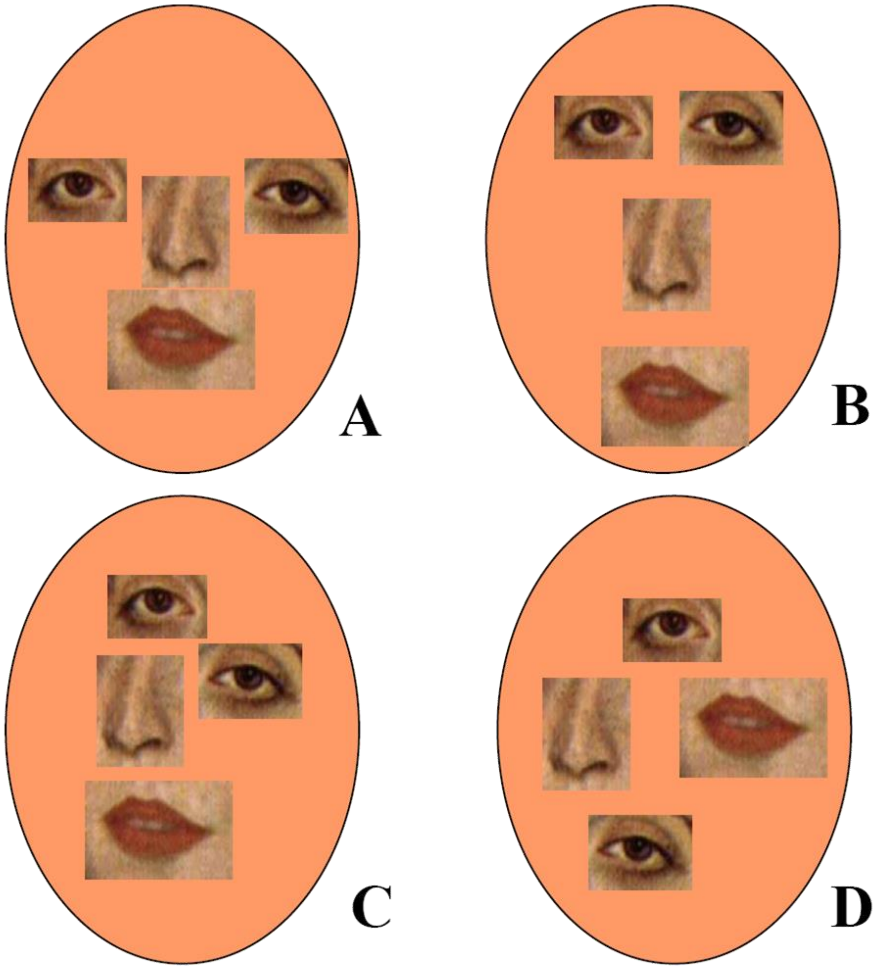
- Bag of words representations make it possible to describe the unordered point set with a single vector (of fixed dimension across image examples).



- Provides easy way to use distribution of feature types with various learning algorithms requiring vector input.

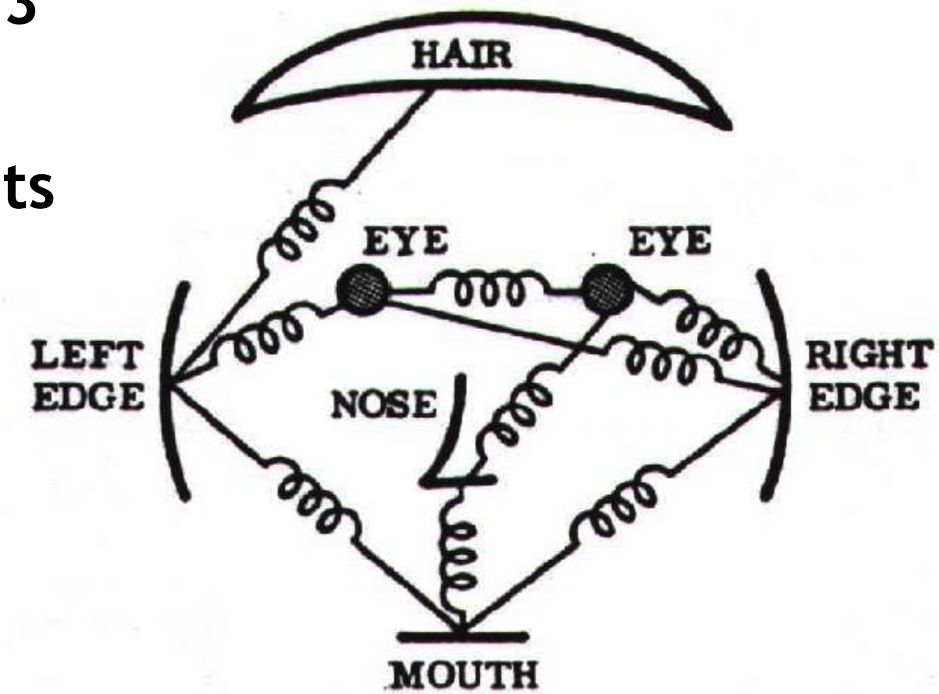
Limitations of BoW Representations

- The bag of words removes spatial layout.
- This is both a strength and a weakness.
- *Why a strength?*
- *Why a weakness?*

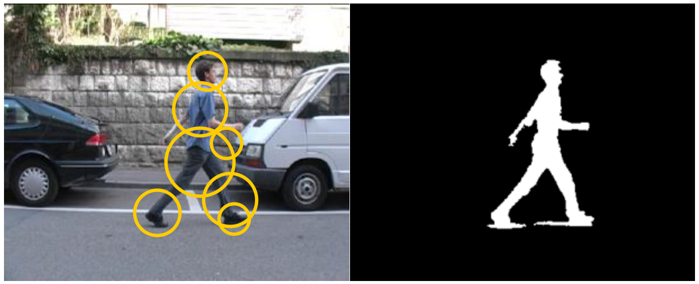


Recap: Part-Based Models

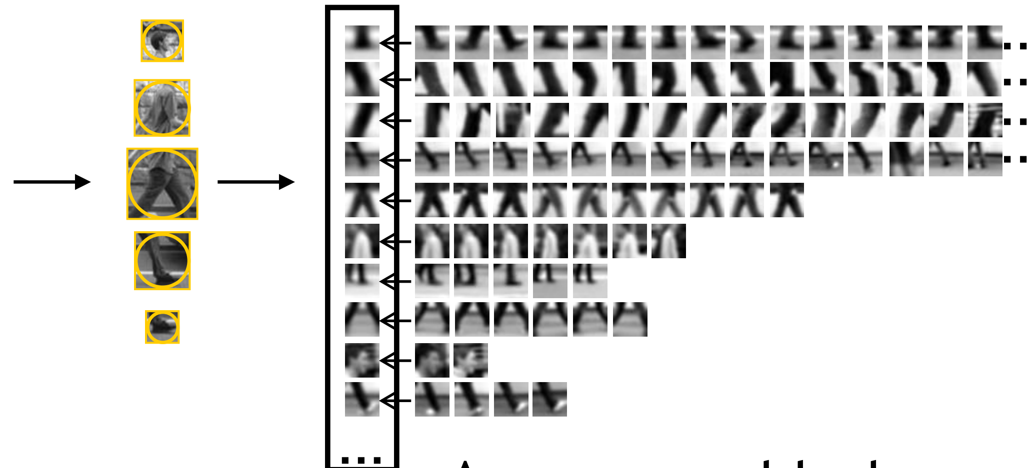
- Fischler & Elschlager 1973
- Model has two components
  - parts
    (2D image fragments)
  - structure
    (configuration of parts)



Recap: Implicit Shape Model - Representation

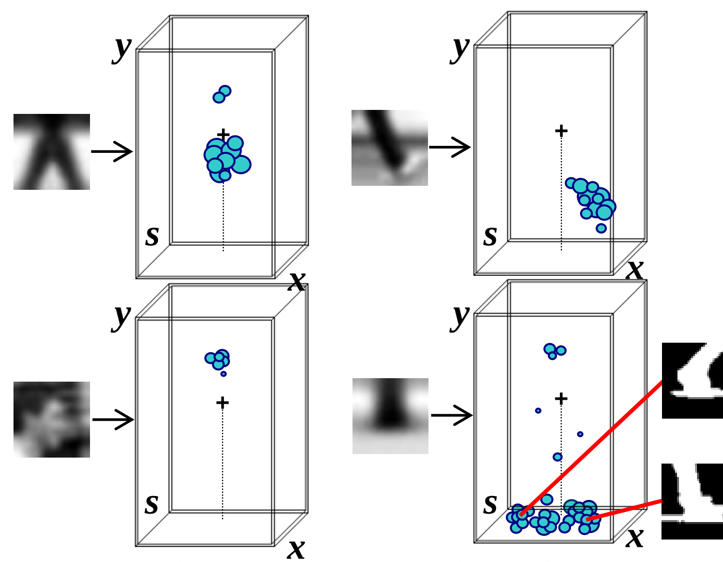


Training images
(+reference segmentation)



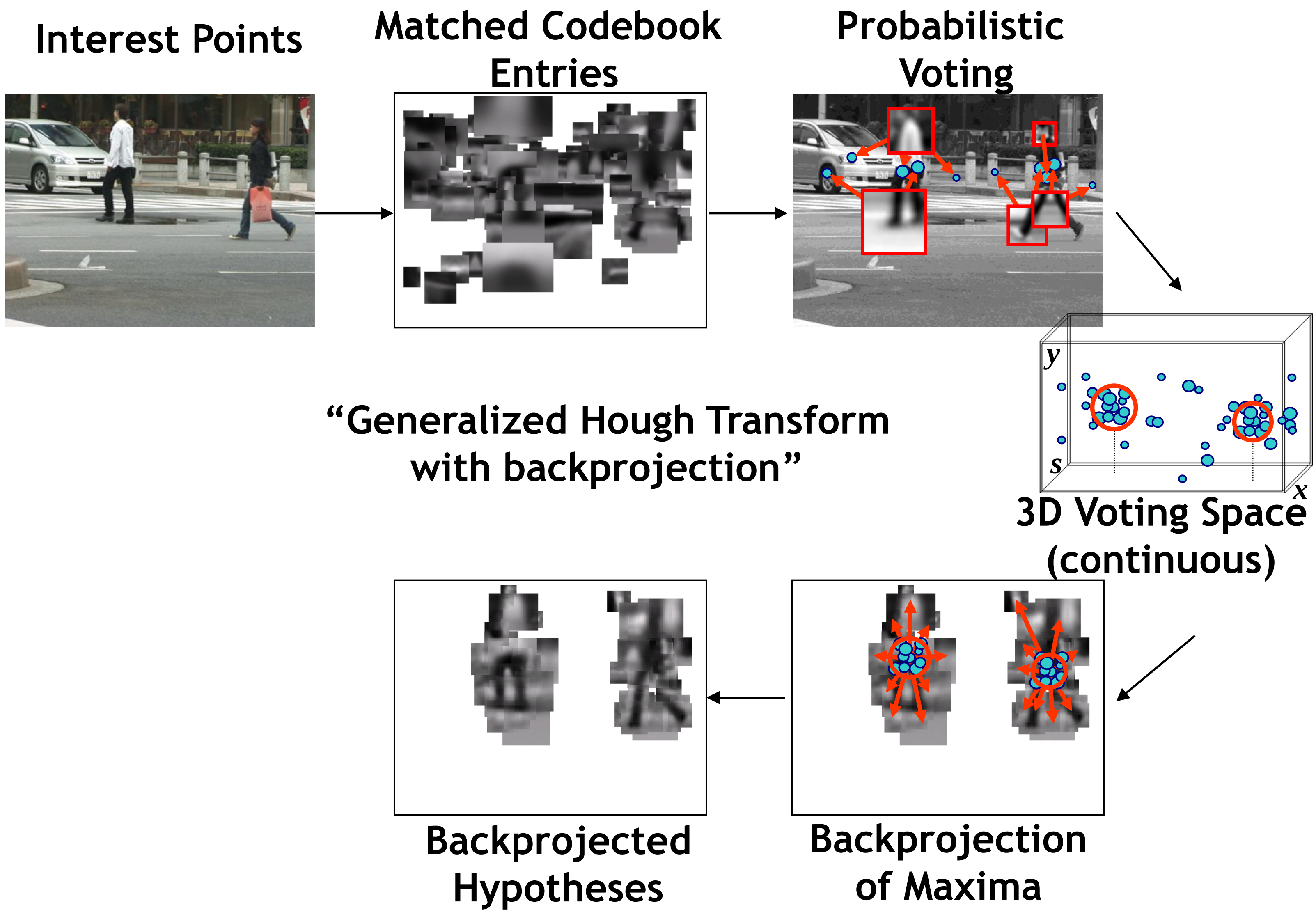
Appearance codebook

- Learn appearance codebook
  - Extract local features at interest points
  - Clustering ⇒ appearance codebook
- Learn spatial distributions
  - Match codebook to training images
  - Record matching positions on object



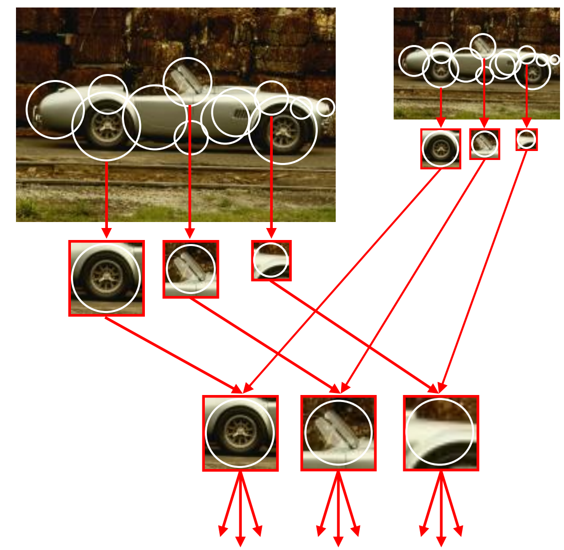
Spatial occurrence distributions
+ local figure-ground labels

Recap: Implicit Shape Model – Recognition



[Leibe, Leonardis, Schiele, SLCV'04; IJCV'08]

Recap: Scale Invariant Voting

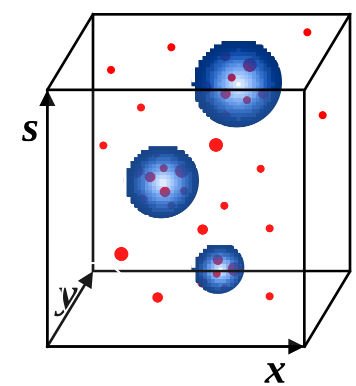


- Scale-invariant feature selection
  - Scale-invariant interest points
  - Rescale extracted patches
  - Match to constant-size codebook

- Generate scale votes
  - Scale as 3rd dimension in voting space

$$
\begin{aligned}
x_{vote} &= x_{img} - x_{occ}(s_{img}/s_{occ}) \\
y_{vote} &= y_{img} - y_{occ}(s_{img}/s_{occ}) \\
s_{vote} &= (s_{img}/s_{occ}).
\end{aligned}
$$

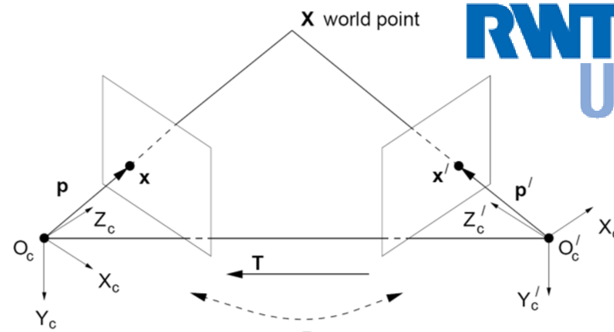
  - Search for maxima in 3D voting space



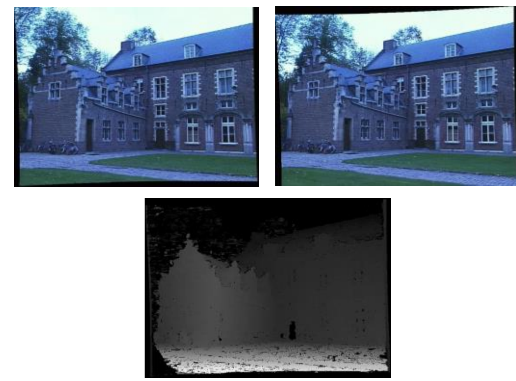
Search window

Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
  - Epipolar Geometry and Stereo Basics
  - Camera Calibration & Uncalibrated Reconstruction
  - Structure-from-Motion
- Motion and Tracking



Epipolar geometry



Disparity

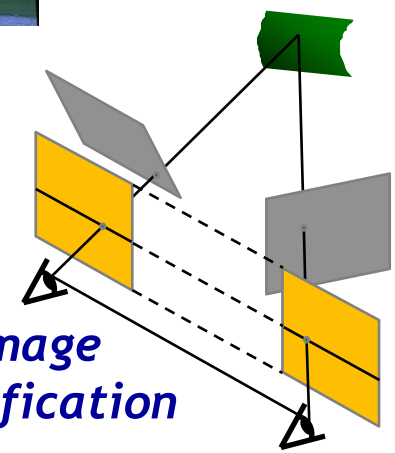
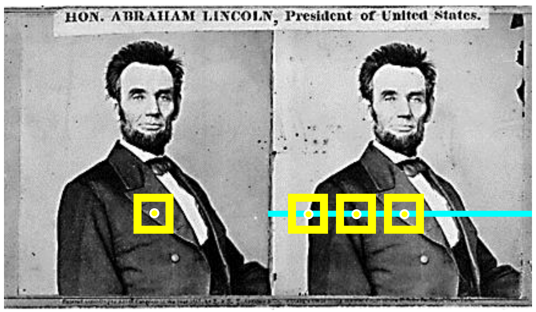


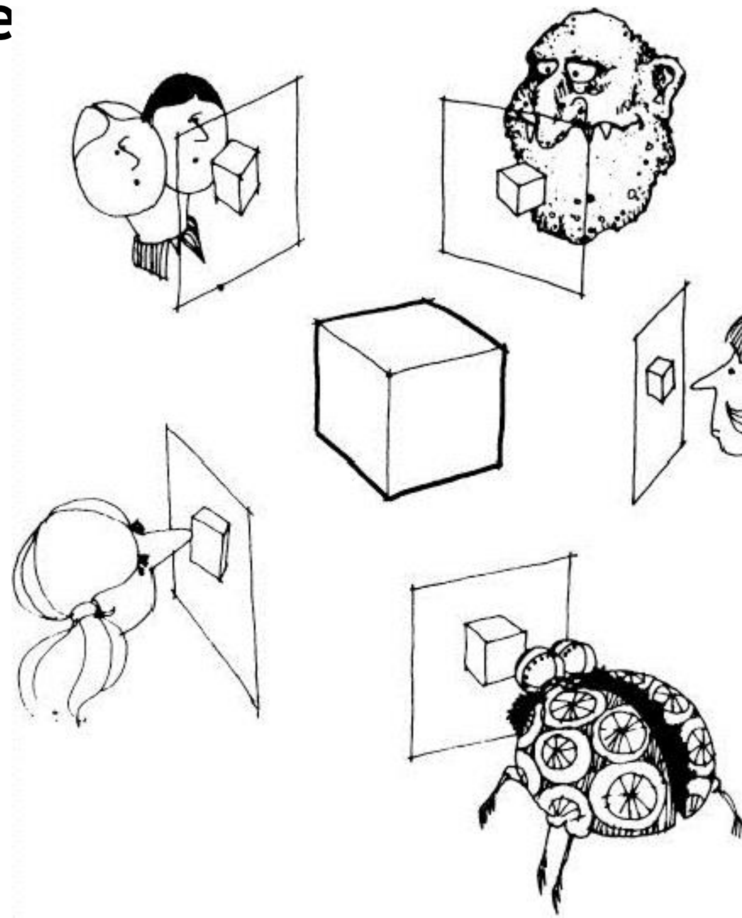
Image rectification



Dense stereo matching

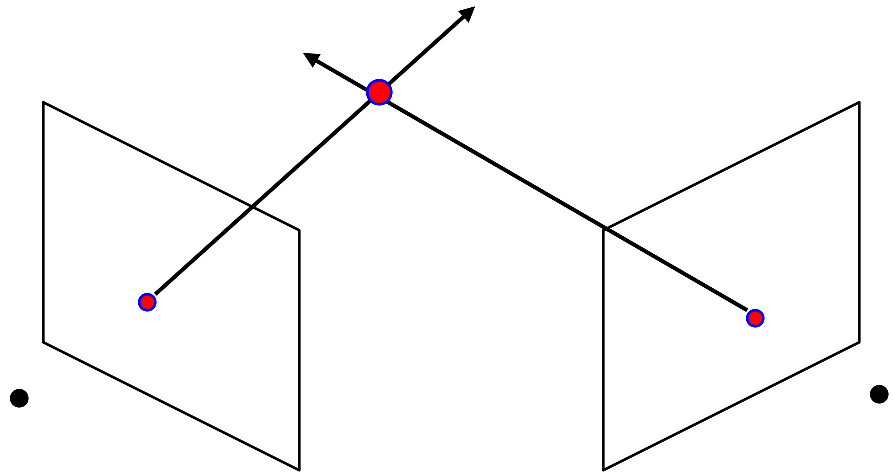
Recap: What Is Stereo Vision?

- **Generic problem formulation: given several images of the same object or scene, compute a representation of its 3D shape**



Slide credit: Svetlana Lazebnik, Steve Seitz

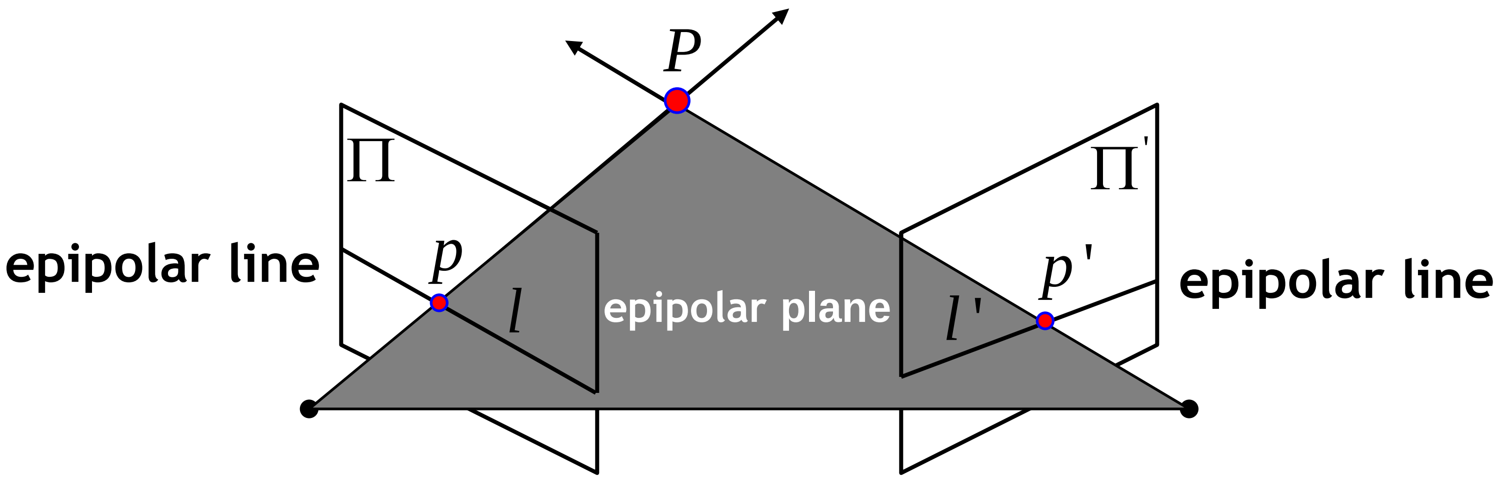
Recap: Depth with Stereo – Basic Idea



- Basic Principle: Triangulation
  - Gives reconstruction as intersection of two rays
  - Requires
    - Camera pose (calibration)
    - Point correspondence

Recap: Epipolar Geometry

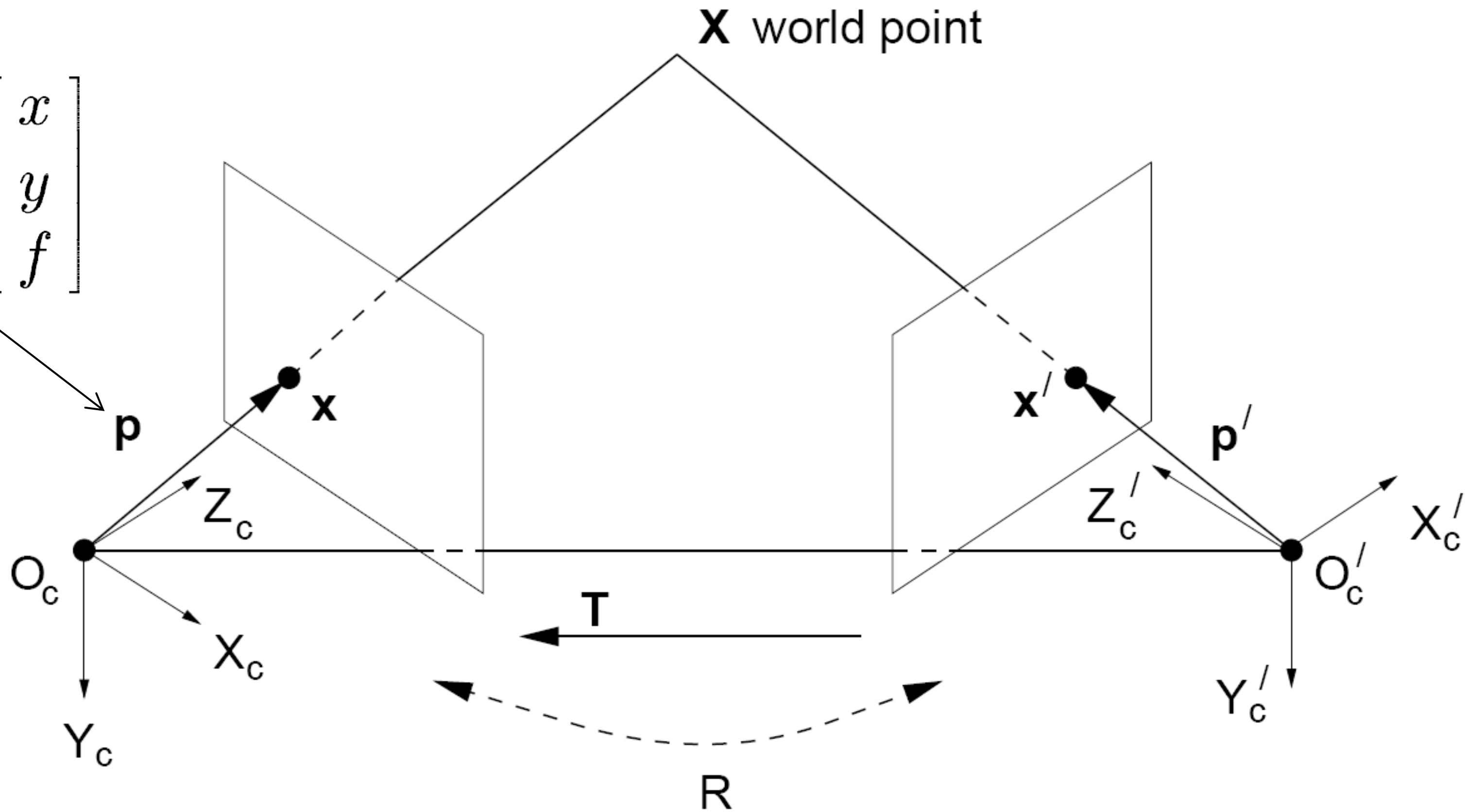
- Geometry of two views allows us to constrain where the corresponding pixel for some image point in the first view must occur in the second view.



- Epipolar constraint:
  - Correspondence for point p in Π must lie on the epipolar line l' in Π' (and vice versa).
  - Reduces correspondence problem to 1D search along conjugate epipolar lines.

Recap: Stereo Geometry With Calibrated Cameras

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ f \end{bmatrix}$$



- Camera-centered coordinate systems are related by known rotation R and translation T:

$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T}$$

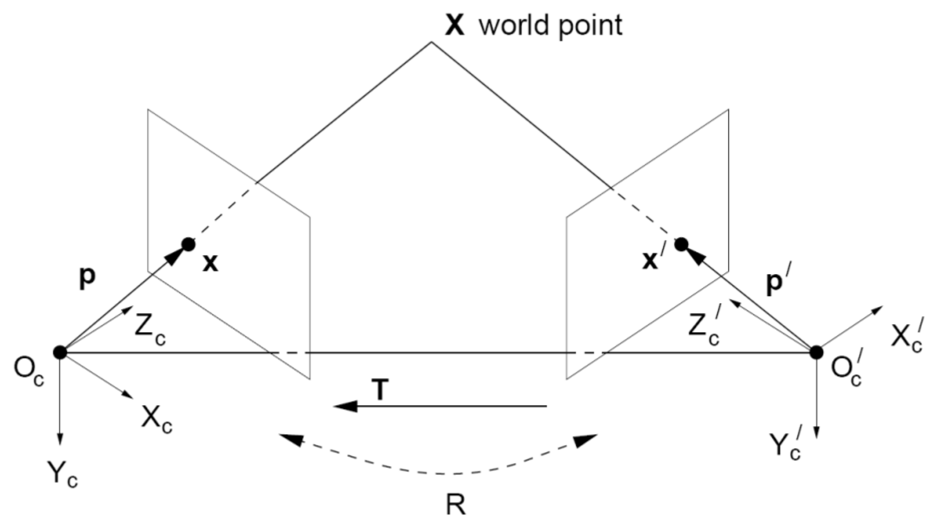
Recap: Essential Matrix

$$\mathbf{X}' \cdot (\mathbf{T} \times \mathbf{R}\mathbf{X}) = 0$$

$$\mathbf{X}' \cdot (\mathbf{T}_x \ \mathbf{R}\mathbf{X}) = 0$$

Let $\mathbf{E} = \mathbf{T}_x\mathbf{R}$

$$\mathbf{X}'^T \mathbf{E} \mathbf{X} = 0$$



- This holds for the rays p and p' that are parallel to the camera-centered position vectors X and X', so we have:

$$\mathbf{p}'^T \mathbf{E}\mathbf{p} = 0$$

- $\mathbf{E}$ is called the essential matrix, which relates corresponding image points [Longuet-Higgins 1981]

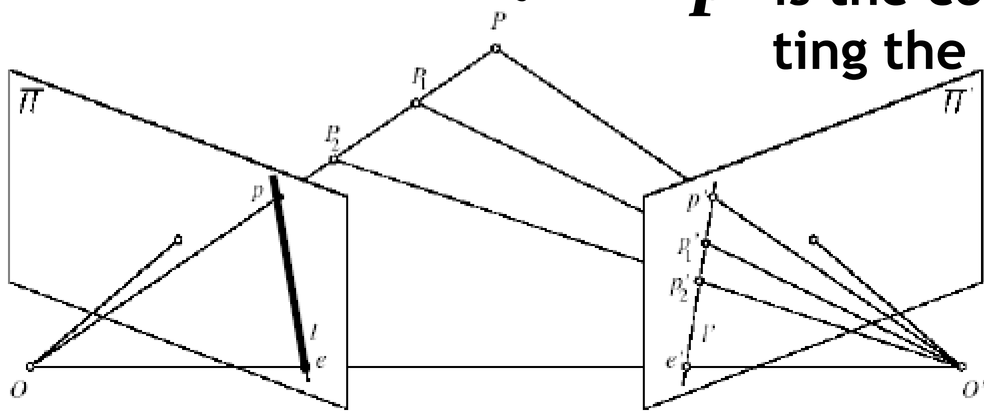
Recap: Essential Matrix and Epipolar Lines

$$\mathbf{p}'^{\mathrm{T}}\mathbf{E}\mathbf{p} = 0$$

Epipolar constraint: if we observe point p in one image, then its position p' in second image must satisfy this equation.

$l' = Ep$ **is the coordinate vector representing the epipolar line for point p**

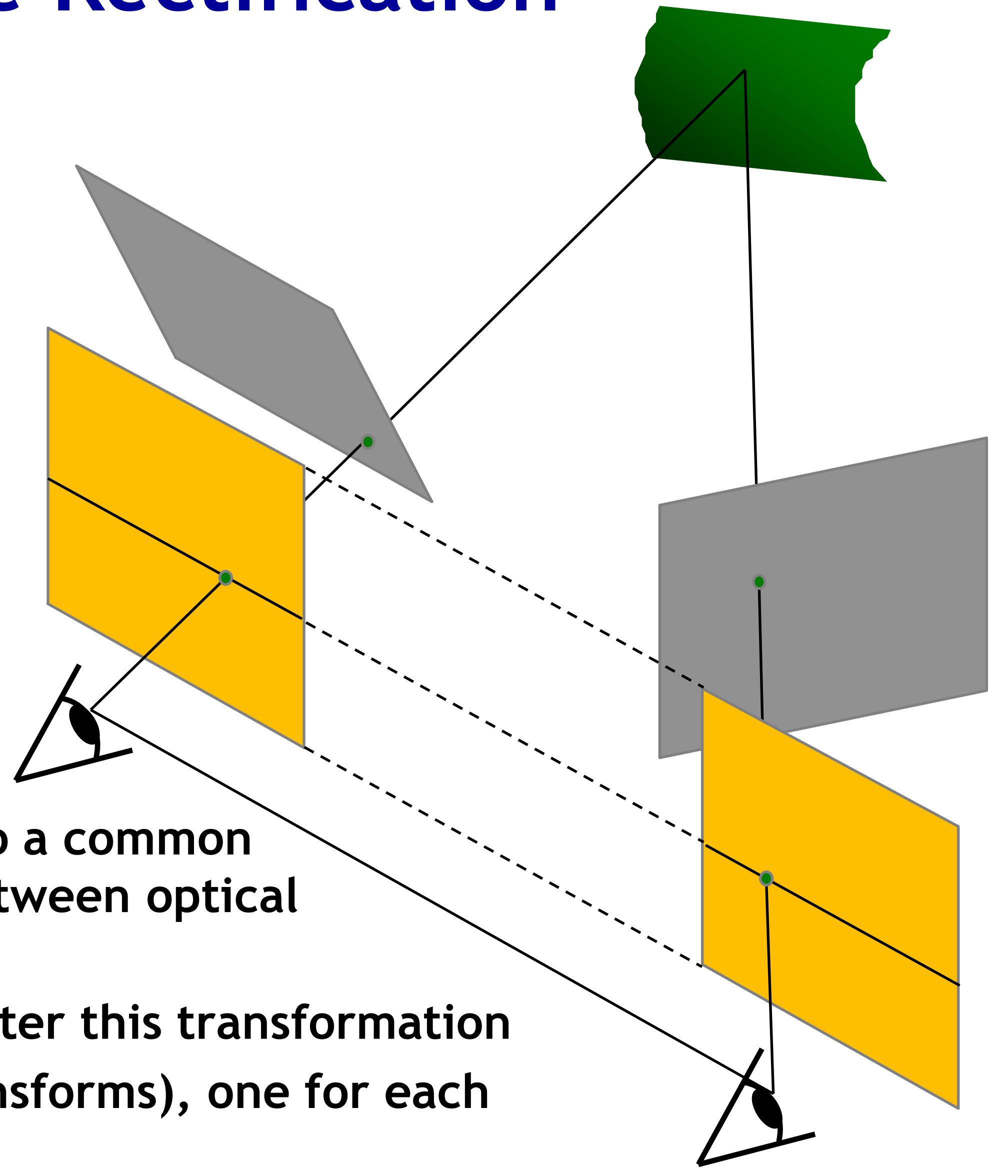
(i.e., the line is given by: $l'^{\top}\mathbf{x} = 0$)



$l = E^{T}p'$ **is the coordinate vector representing the epipolar line for point p'**

Recap: Stereo Image Rectification

- In practice, it is convenient if image scanlines are the epipolar lines.

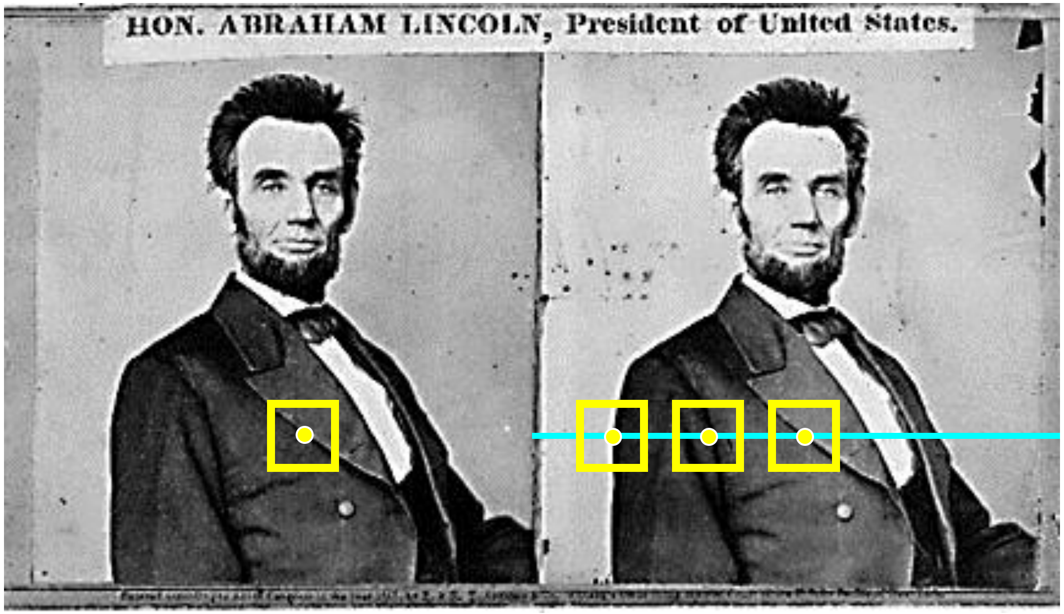


- Algorithm
  - Reproject image planes onto a common plane parallel to the line between optical centers
  - Pixel motion is horizontal after this transformation
  - Two homographies (3x3 transforms), one for each input image reprojection

Slide adapted from Li Zhang

C. Loop & Z. Zhang, Computing Rectifying Homographies for Stereo Vision. CVPR'99

Recap: Dense Correspondence Search

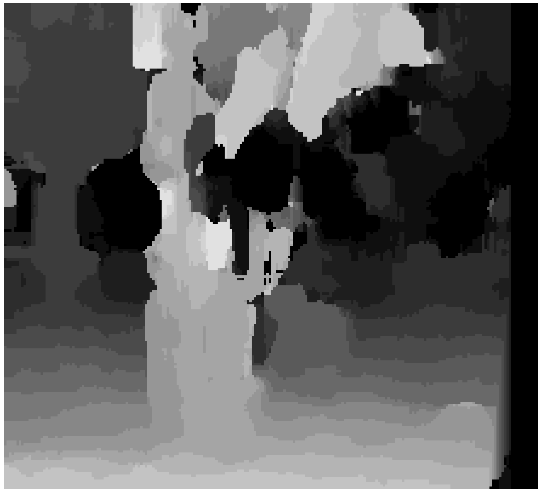


- For each pixel in the first image
  - Find corresponding epipolar line in the right image
  - Examine all pixels on the epipolar line and pick the best match (e.g. SSD, correlation)
  - Triangulate the matches to get depth information
- This is easiest when epipolar lines are scanlines
  - ⇒ Rectify images first

adapted from Svetlana Lazebnik, Li Zhang

Recap: Effect of Window Size

W = 3



W = 20

Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.

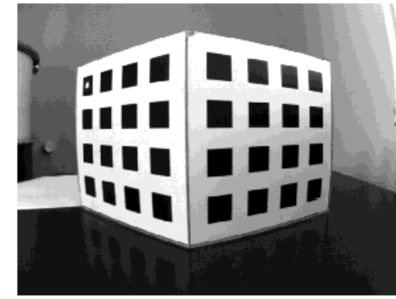
Repetition

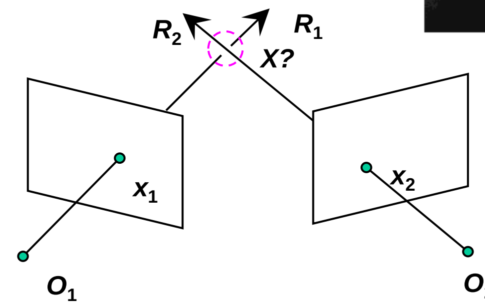
- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
  - Epipolar Geometry and Stereo Basics
  - Camera Calibration & Uncalibrated Reconstruction
  - Structure-from-Motion
- Motion and Tracking

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}$$

Camera models

Camera calibration





Triangulation

Essential matrix, Fundamental matrix

$$x^T E x' = 0$$

$$x^T F x' = 0$$

$$\begin{bmatrix}
u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\
u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' & 1 \\
u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' & 1 \\
u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' & 1 \\
u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' & 1 \\
u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' & 1 \\
u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' & 1 \\
u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' & 1
\end{bmatrix}
\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

Eight-point algorithm

SVD!

Recap: A General Point

- Equations of the form

$$\mathrm{Ax} = 0$$

- How do we solve them? (always!)
  - Apply SVD

$$\boldsymbol{A} \overset{\text{SVD}}{=} \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T = \boldsymbol{U}\begin{bmatrix} d_{11} & & \\ & \ddots & \\ & & d_{NN} \end{bmatrix}\begin{bmatrix} v_{11} & \cdots & v_{1N} \\ \vdots & \ddots & \vdots \\ v_{N1} & \cdots & v_{NN} \end{bmatrix}^T$$

Singular values   Singular vectors

  - Singular values of A = square roots of the eigenvalues of A^TA.
  - The solution of Ax=0 is the *nullspace* vector of A.
  - This corresponds to the *smallest singular vector* of A.

Recap: Camera Parameters

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*

$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & s & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & s' & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}$$

- Extrinsic parameters
  - Rotation R
  - Translation t
    (both relative to world coordinate system)

- Camera projection matrix

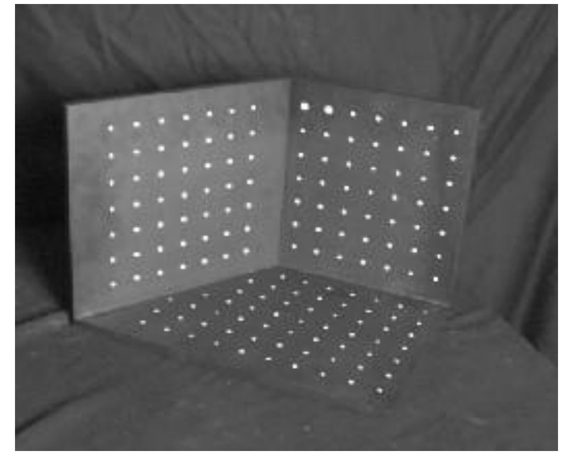
$$\mathrm{P} = \mathrm{K}\left[\mathrm{R} \mid \mathrm{t}\right]$$

$\Rightarrow$ General pinhole camera: 9 DoF

$\Rightarrow$ CCD Camera with square pixels: 10 DoF

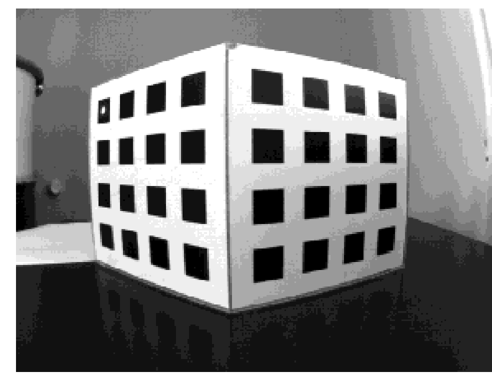
$\Rightarrow$ General camera: 11 DoF

Recap: Calibrating a Camera



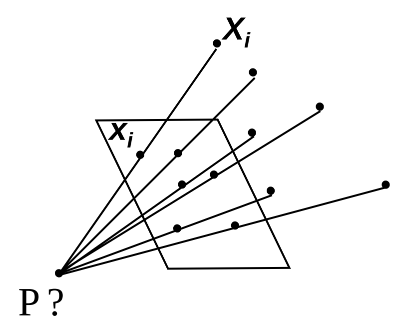
Goal

- Compute intrinsic and extrinsic parameters using observed camera data.



Main idea

- Place "calibration object" with known geometry in the scene
- Get correspondences
- Solve for mapping from scene to image: estimate $P = P_{int} P_{ext}$



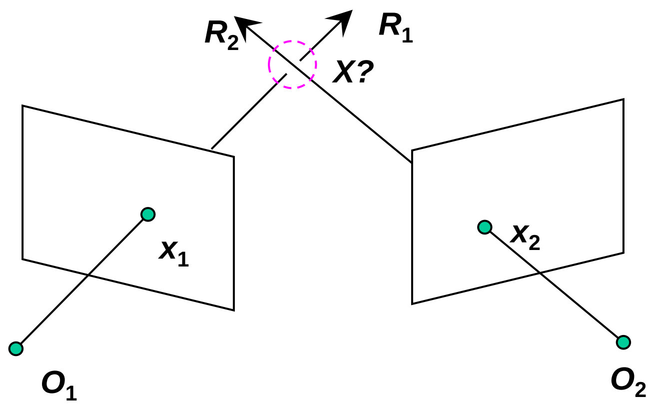
Recap: Camera Calibration (DLT Algorithm)

$$\begin{bmatrix} 0^T & X_1^T & -y_1 X_1^T \\ X_1^T & 0^T & -x_1 X_1^T \\ \cdots & \cdots & \cdots \\ 0^T & X_n^T & -y_n X_n^T \\ X_n^T & 0^T & -x_n X_n^T \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix} = 0 \qquad \mathrm{Ap} = 0$$

- **P has 11 degrees of freedom.**
- **Two linearly independent equations per independent 2D/3D correspondence.**
- Solve with SVD (similar to homography estimation)
  - Solution corresponds to smallest singular vector.
- **5 ½ correspondences needed for a minimal solution.**

Recap: Triangulation - Lin. Alg. Approach

see Exercise 6.3!



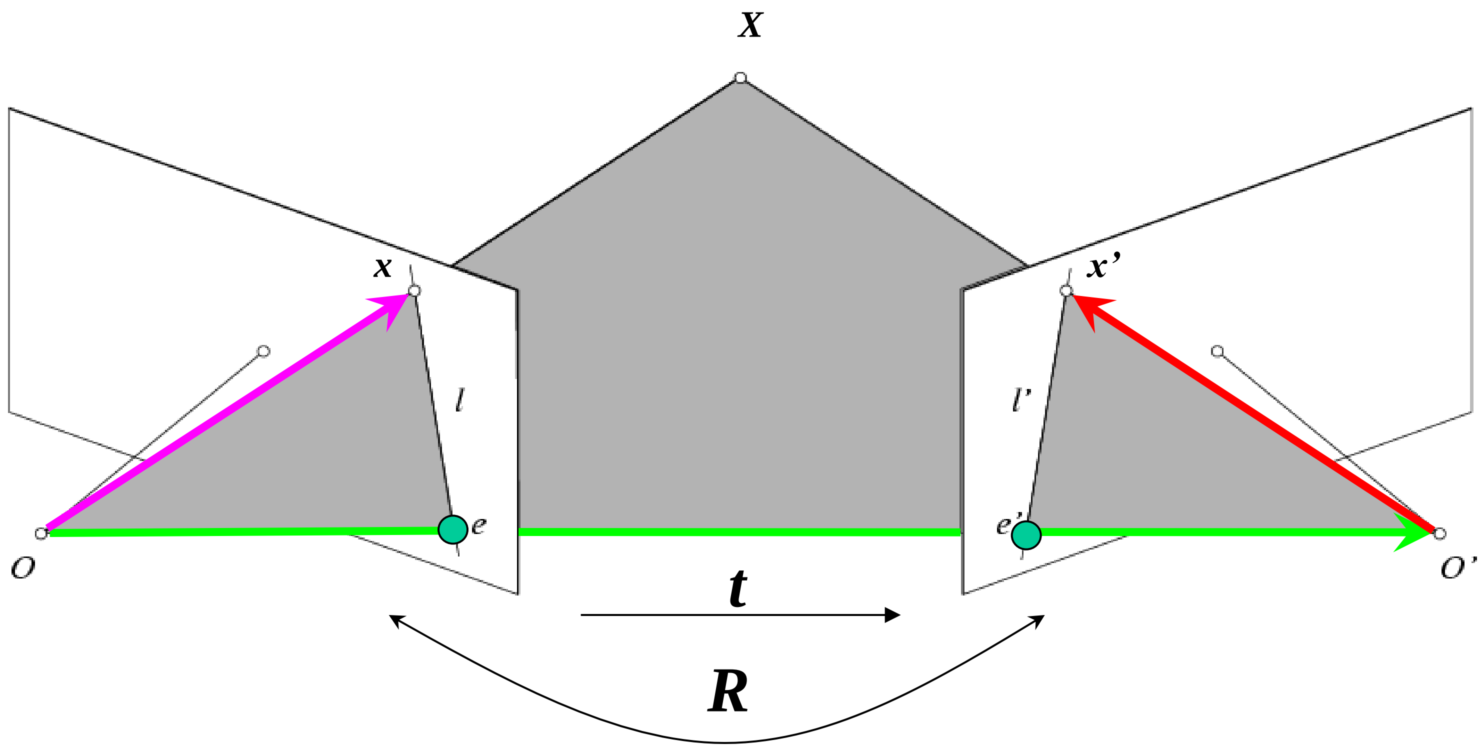
$$\lambda_1 x_1 = P_1 X \qquad x_1 \times P_1 X = 0 \qquad [x_{1\times}] P_1 X = 0$$

$$\lambda_2 x_2 = P_2 X \qquad x_2 \times P_2 X = 0 \qquad [x_{2\times}] P_2 X = 0$$

- **Two independent equations each in terms of three unknown entries of X.**
- **Stack equations and solve with SVD.**
- **This approach nicely generalizes to multiple cameras.**

Slide credit: Svetlana Lazebnik

Recap: Epipolar Geometry – Calibrated Case



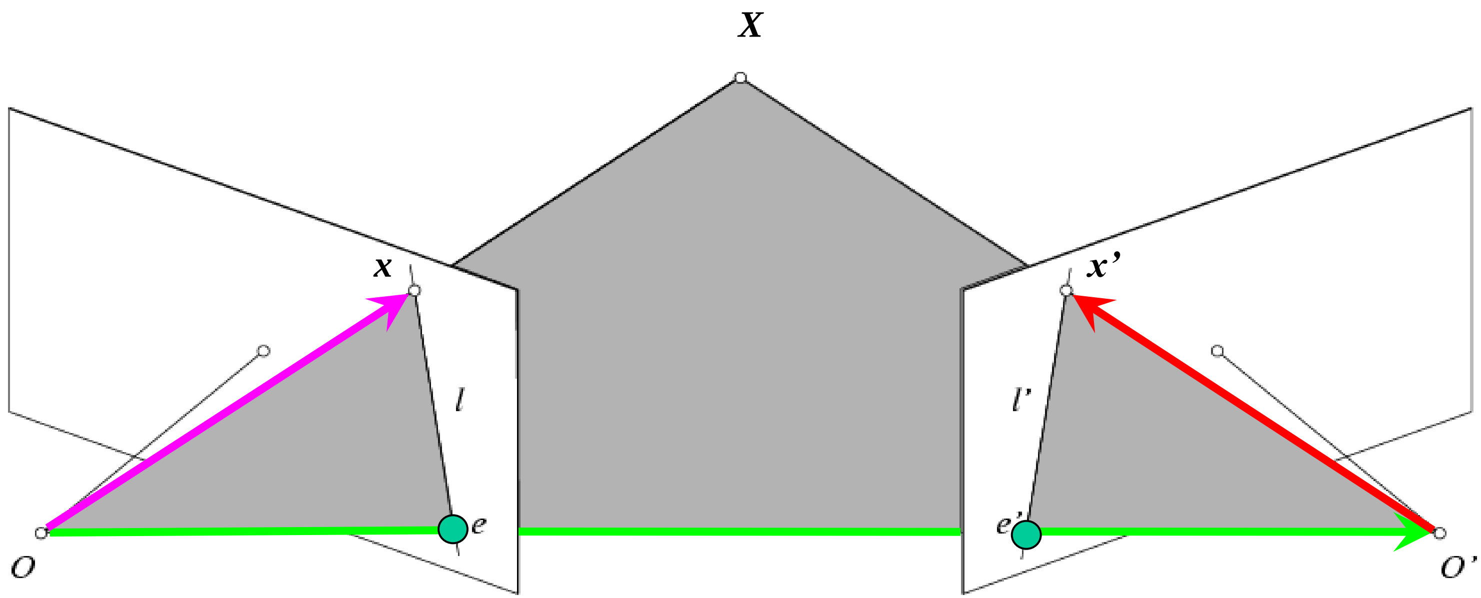
Camera matrix: [I|0]
$X = (u, v, w, 1)^T$
$x = (u, v, w)^T$

Camera matrix: $[R^T \mid -R^T t]$
Vector x' in second coord. system has coordinates Rx' in the first one.

The vectors x, t, and Rx' are coplanar

Slide credit: Svetlana Lazebnik

Recap: Epipolar Geometry – Calibrated Case

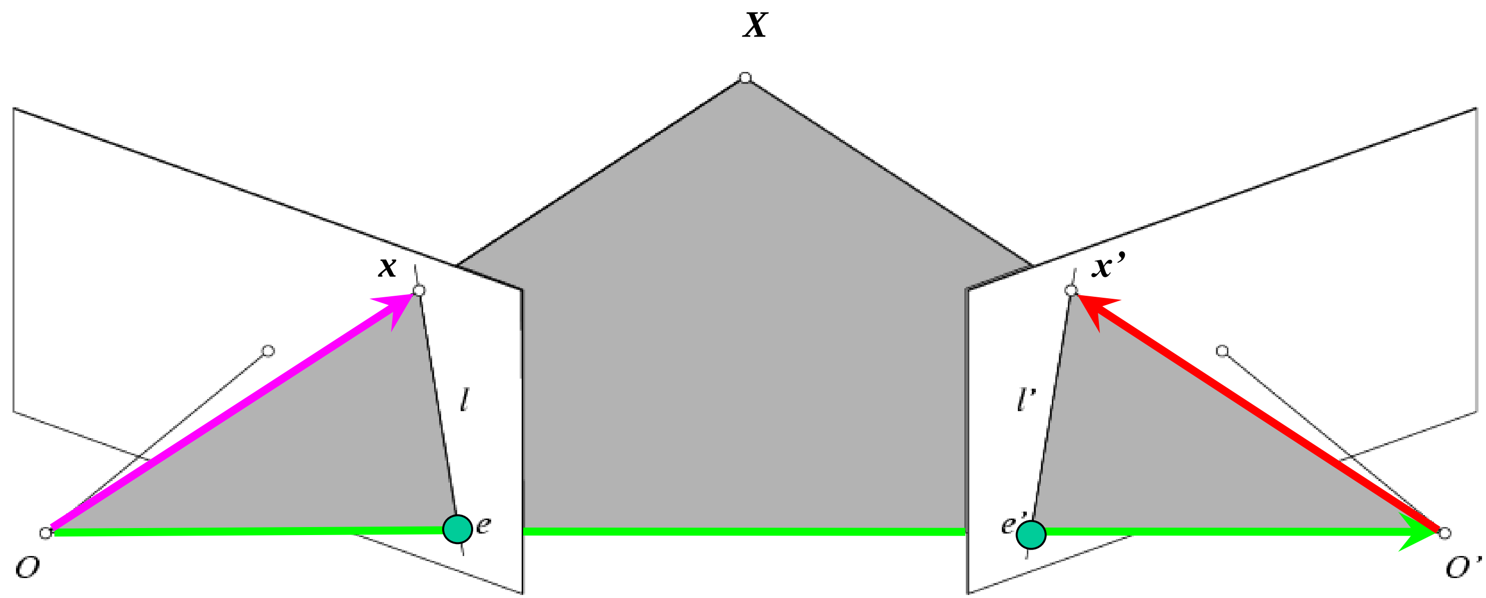


$$x \cdot [t \times (Rx')] = 0 \quad \Rightarrow \quad x^T E x' = 0 \quad \text{with} \quad E = [t_\times]R$$

Essential Matrix (Longuet-Higgins, 1981)

RWTH AACHEN UNIVERSITY

Recap: Epipolar Geometry – Calibrated Case

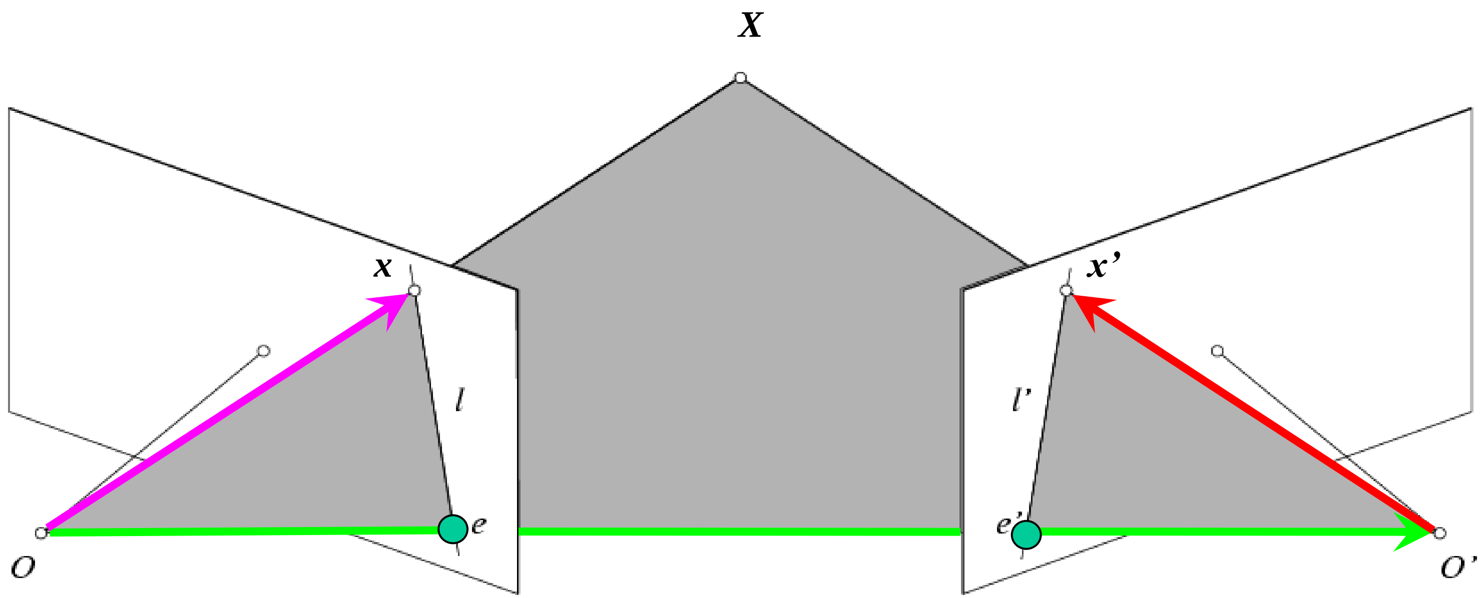


$$x \cdot [t \times (Rx')] = 0 \quad \Rightarrow \quad x^T E x' = 0 \quad \text{with} \quad E = [t_\times]R$$

- ***E x'*** is the epipolar line associated with ***x'*** (***l*** = ***E x'***)
- ***E***^{T}***x*** is the epipolar line associated with ***x*** (***l'*** = ***E***^{T}***x***)
- ***E e'*** = 0 and ***E***^{T}***e*** = ***0***
- ***E*** is singular (rank two)
- ***E*** has five degrees of freedom (up to scale)

Slide credit: Svetlana Lazebnik

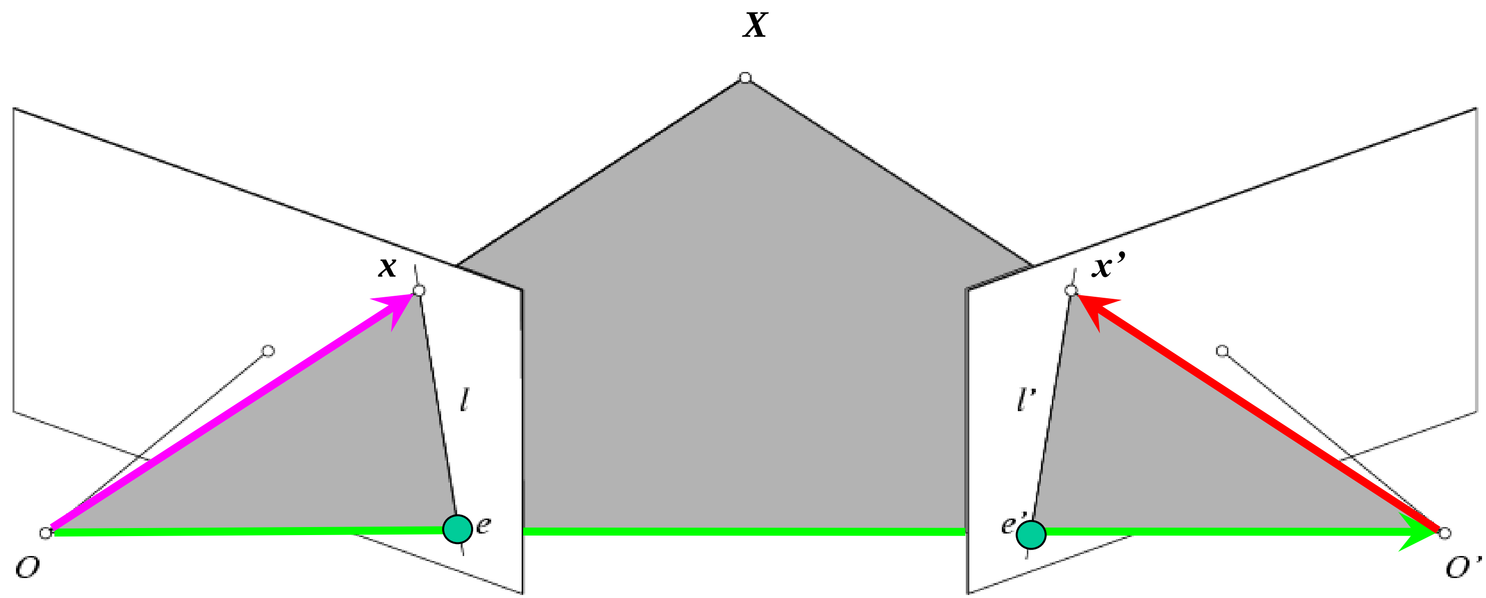
Recap: Epipolar Geometry - Uncalibrated Case



- **The calibration matrices *K* and *K'* of the two cameras are unknown**
- **We can write the epipolar constraint in terms of *unknown* normalized coordinates:**

$$\hat{x}^T E \hat{x}' = 0 \qquad\qquad x = K\hat{x}, \quad x' = K'\hat{x}'$$

Recap: Epipolar Geometry - Uncalibrated Case



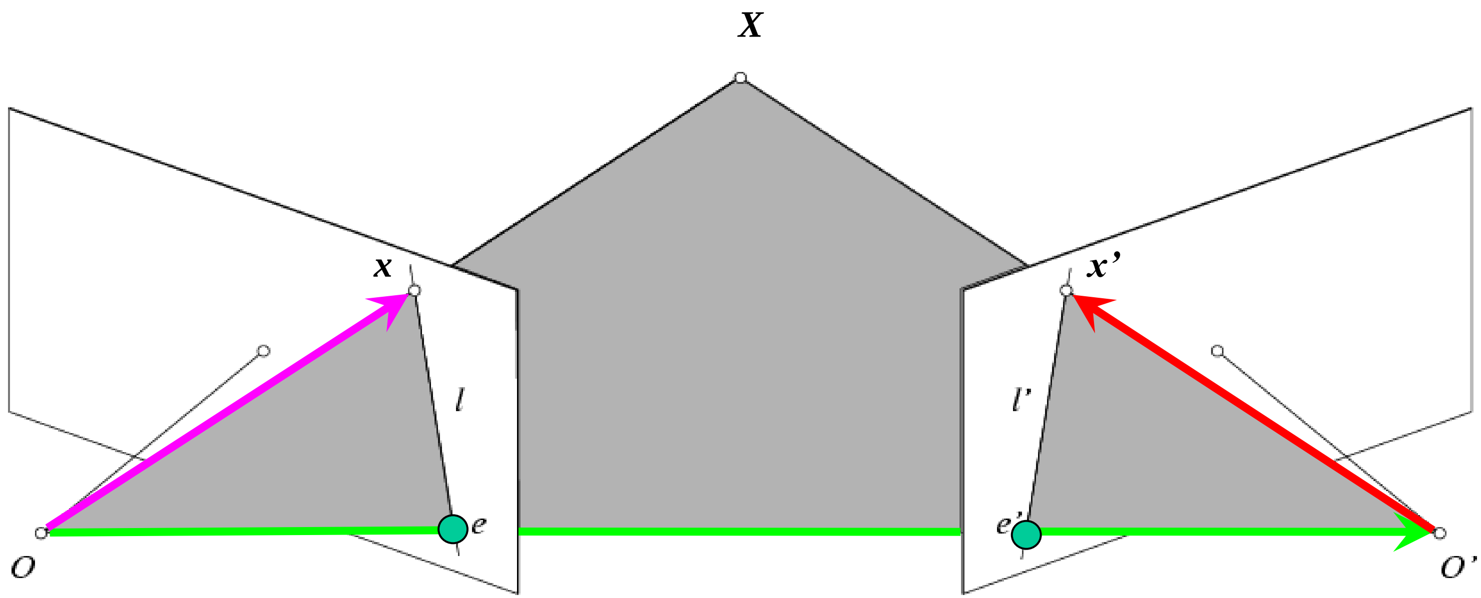
$$\hat{x}^T E \hat{x}' = 0 \Rightarrow x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

$$x = K \hat{x}$$

$$x' = K' \hat{x}'$$

Fundamental Matrix
(Faugeras and Luong, 1992)

Recap: Epipolar Geometry - Uncalibrated Case



$$\hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

- ***F x'*** is the epipolar line associated with ***x'*** (***l*** = ***F x'***)
- ***F^{T}x*** is the epipolar line associated with ***x*** (***l'*** = ***F^{T}x***)
- ***F e'*** = 0 and ***F^{T}e*** = ***0***
- ***F*** is singular (rank two)
- ***F*** has seven degrees of freedom

Recap: The Eight-Point Algorithm

see Exercise 6.1!

$\mathbf{x} = (u, v, 1)^T, \quad \mathbf{x'} = (u', v', 1)^T$

$$(u, v, 1)\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0 \quad \Rightarrow \quad (uu', uv', u, vu', vv', v, u', v', 1)\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = 0$$

$$\begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' & 1 \\ u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' & 1 \\ u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' & 1 \\ u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' & 1 \\ u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' & 1 \\ u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' & 1 \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' & 1 \end{bmatrix}\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

1.) Solve with SVD. This minimizes

$$\sum_{i=1}^{N} (x_i^T F x_i')^2$$

2.) Enfore rank-2 constraint using SVD

- **Problem: poor numerical conditioning**

Recap: Normalized Eight-Point Alg.

see Exercise 6.1!

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.
2. Use the eight-point algorithm to compute *F* from the normalized points.
3. Enforce the rank-2 constraint using SVD.

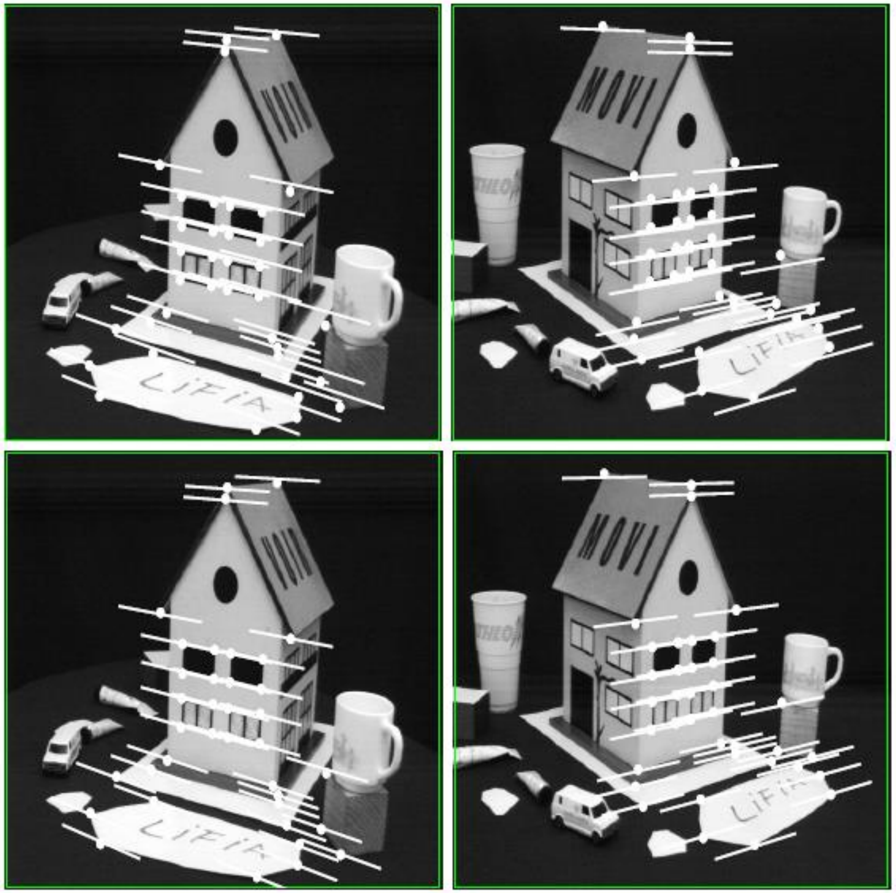
$$F \overset{\text{SVD}}{=} UDV^T = U \begin{bmatrix} d_{11} & & \\ & d_{22} & \\ & & d_{33} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{13} \\ \vdots & \ddots & \vdots \\ v_{31} & \cdots & v_{33} \end{bmatrix}^T$$

Set d_{33} to zero and reconstruct F

4. Transform fundamental matrix back to original units: if T and T' are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is $T^T F T'$.

[Hartley, 1995]

Recap: Comparison of Estimation Algorithms

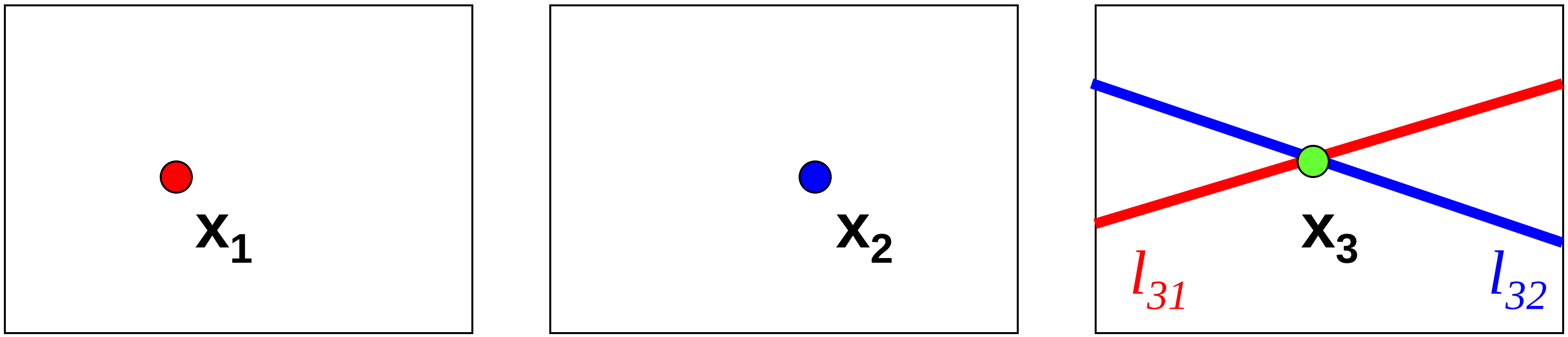


	8-point	Normalized 8-point	Nonlinear least squares
Av. Dist. 1	2.33 pixels	0.92 pixel	0.86 pixel
Av. Dist. 2	2.18 pixels	0.85 pixel	0.80 pixel

Slide credit: Svetlana Lazebnik

Recap: Epipolar Transfer

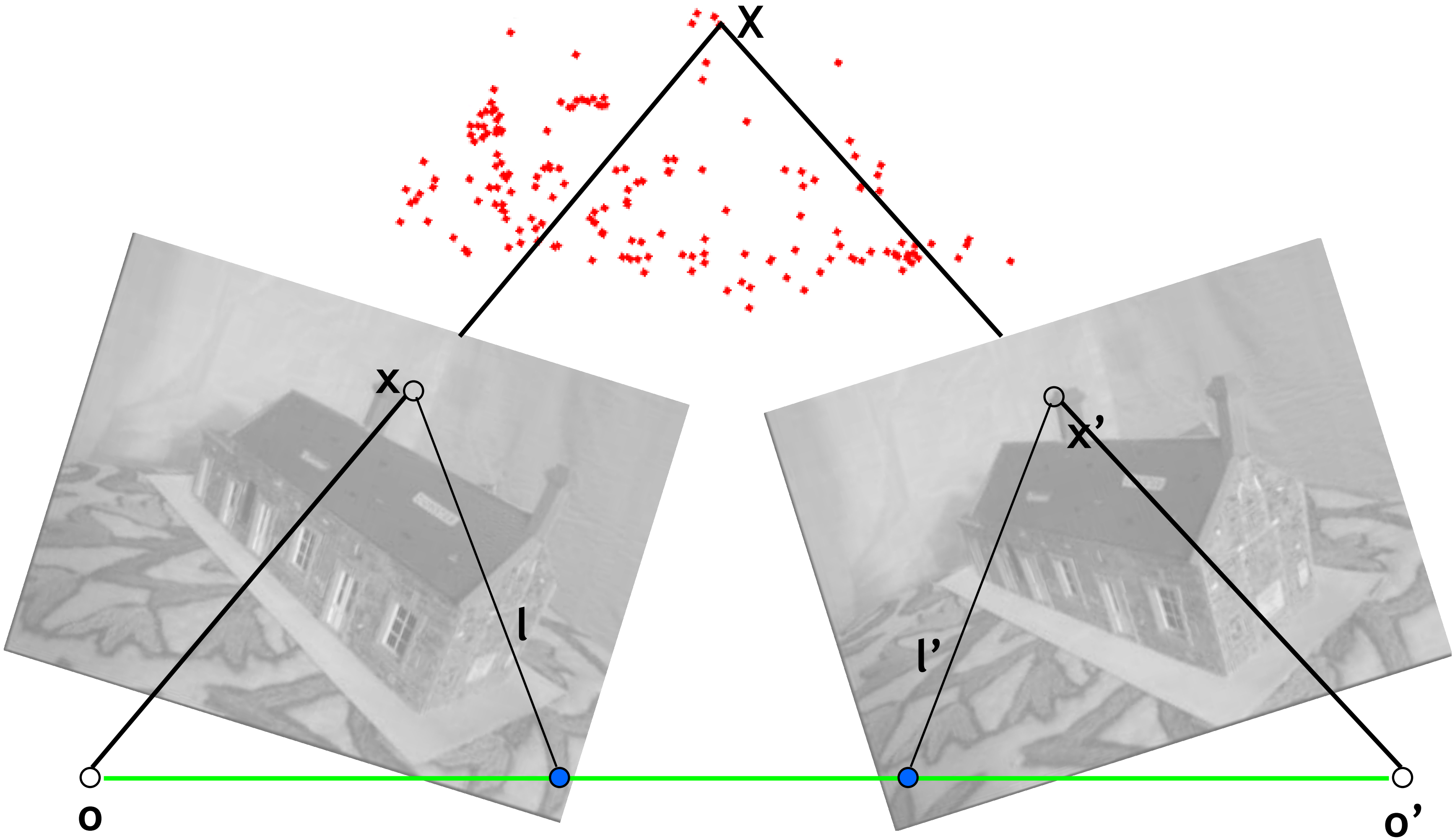
- Assume the epipolar geometry is known
- Given projections of the same point in two images, how can we compute the projection of that point in a third image?



$$l_{31} = F^{T}{}_{13} x_1$$

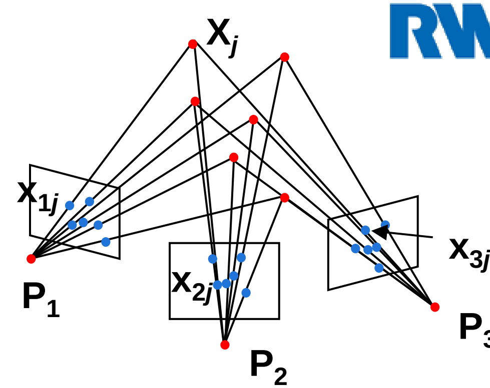
$$l_{32} = F^{T}{}_{23} x_2$$

Applications: 3D Reconstruction



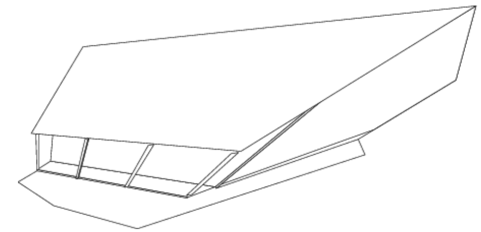
Repetition

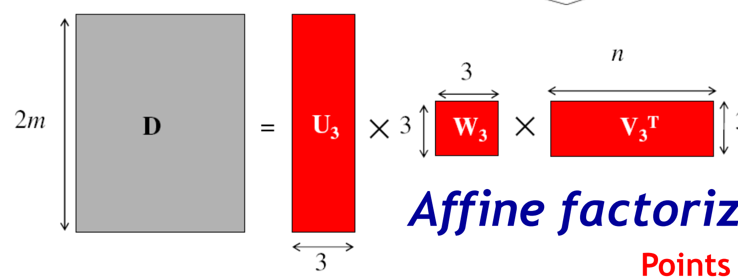
- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
  - Epipolar Geometry and Stereo Basics
  - Camera Calibration & Uncalibrated Reconstruction
  - Structure-from-Motion
- Motion and Tracking



Structure-from-Motion

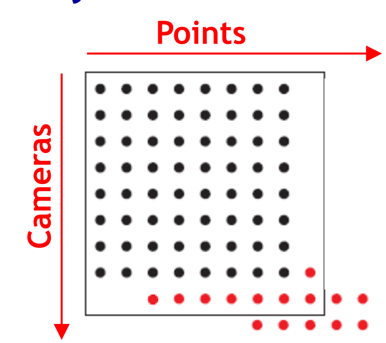
Projective ambiguity

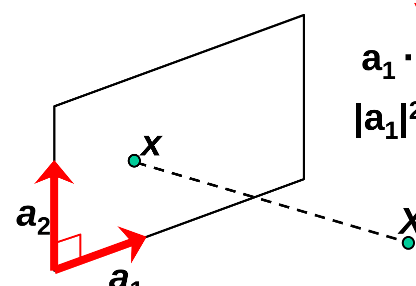




Affine factorization

Projective factorization



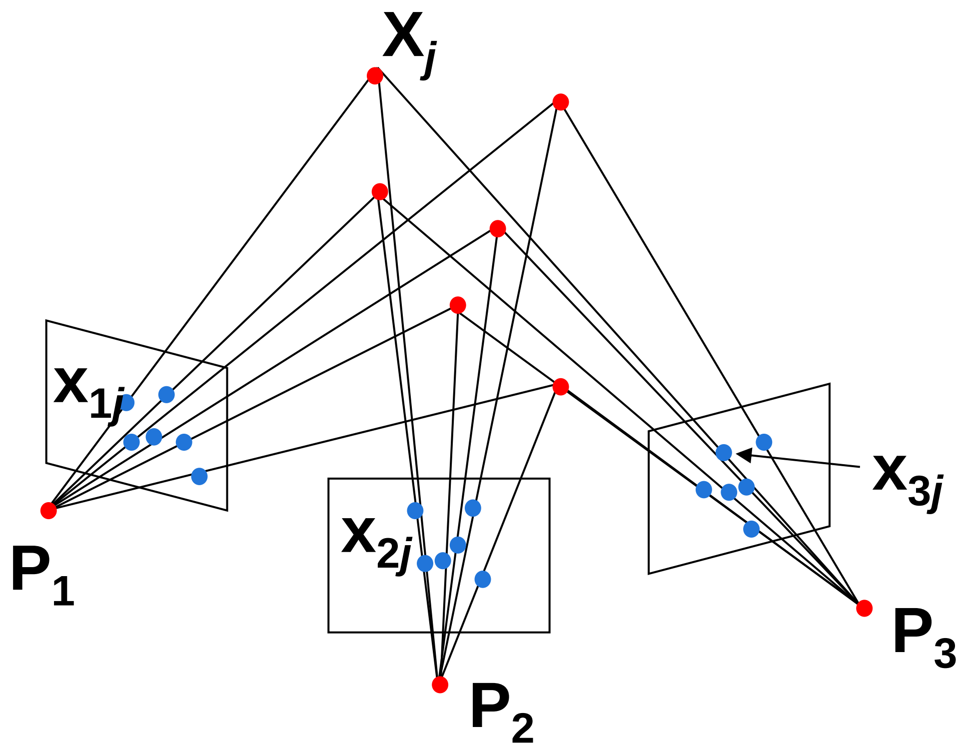


$a_1 \cdot a_2 = 0$

$|a_1|^2 = |a_2|^2 = 1$

Euclidean upgrade

Recap: Structure from Motion



- Given: *m* images of *n* fixed 3D points

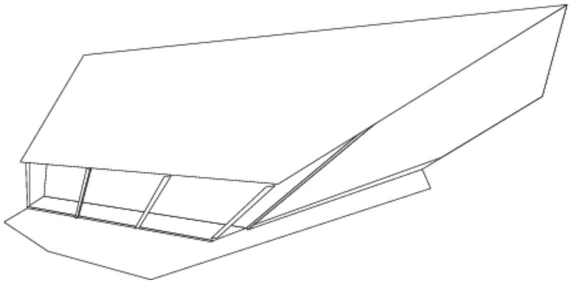
$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \qquad i = 1, \dots, m, \quad j = 1, \dots, n$$

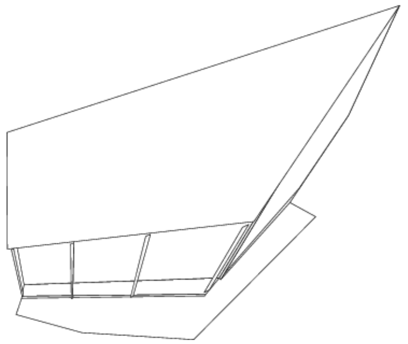
- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$

Recap: Structure from Motion Ambiguity

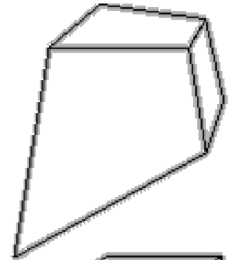
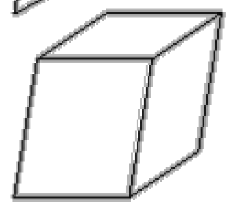
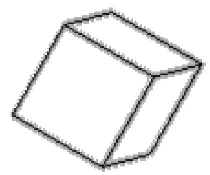
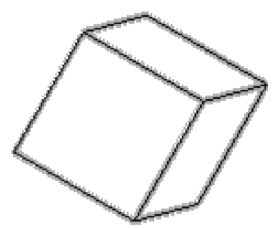
- If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same.
- More generally: if we transform the scene using a transformation $\mathbf{Q}$ and apply the inverse transformation to the camera matrices, then the images do not change

$$\mathbf{x} = \mathbf{P}\mathbf{X} = (\mathbf{P}\mathbf{Q}^{-1})\mathbf{Q}\mathbf{X}$$





Recap: Hierarchy of 3D Transformations

Projective **15dof**	$\begin{bmatrix} A & t \\ v^T & v \end{bmatrix}$		**Preserves intersection and tangency**
Affine **12dof**	$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$		**Preserves parallellism, volume ratios**
Similarity **7dof**	$\begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix}$		**Preserves angles, ratios of length**
Euclidean **6dof**	$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$		**Preserves angles, lengths**

- **With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction.**
- **Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean.**

Recap: Affine Structure from Motion

- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

Points ($3 \times n$)

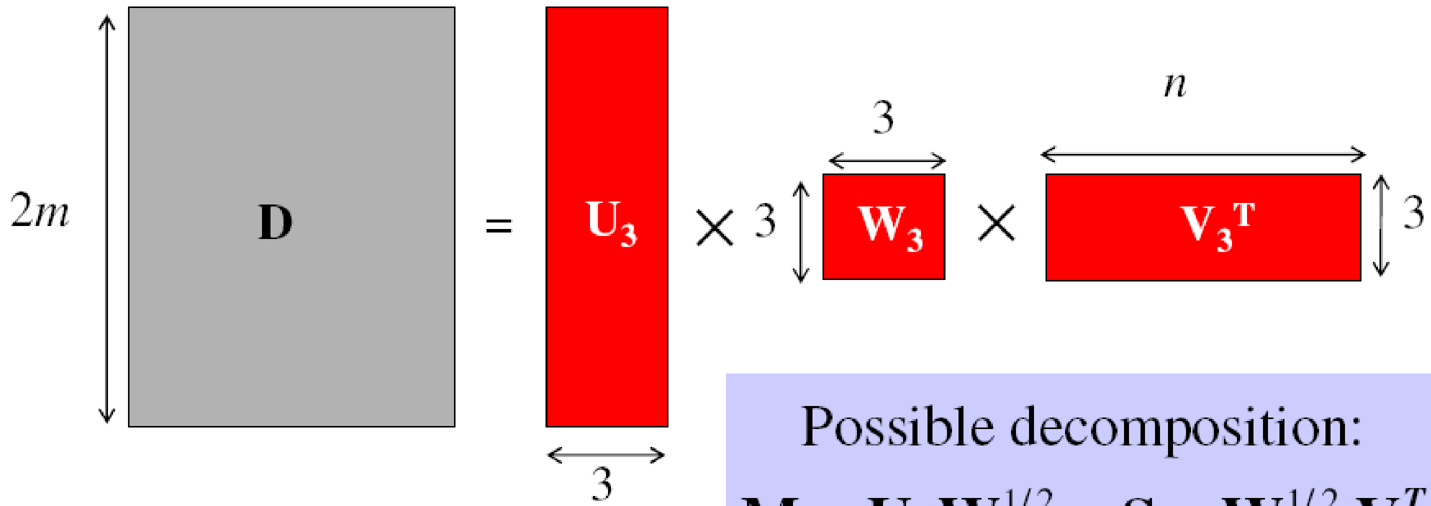
Cameras ($2m \times 3$)

- The measurement matrix $\mathbf{D} = \mathbf{MS}$ must have rank 3!

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

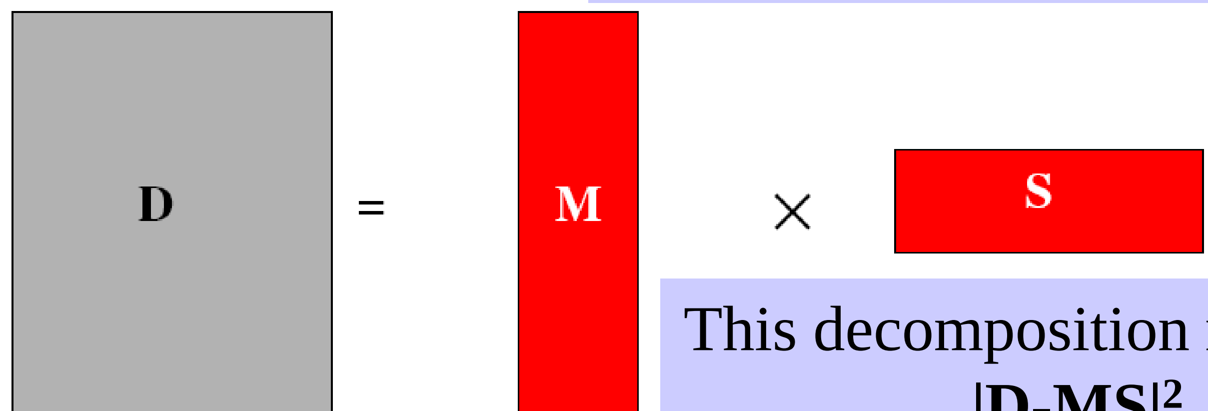
Recap: Affine Factorization

- **Obtaining a factorization from SVD:**



Possible decomposition:

$$\mathbf{M} = \mathbf{U}_3 \mathbf{W}_3^{1/2} \qquad \mathbf{S} = \mathbf{W}_3^{1/2}\, \mathbf{V}_3^T$$



This decomposition minimizes
$|\mathbf{D}\text{-}\mathbf{MS}|^2$

Slide credit: Martial Hebert

Recap: Projective Factorization

$$\mathbf{D} = \begin{bmatrix} z_{11}\mathbf{x}_{11} & z_{12}\mathbf{x}_{12} & \cdots & z_{1n}\mathbf{x}_{1n} \\ z_{21}\mathbf{x}_{21} & z_{22}\mathbf{x}_{22} & \cdots & z_{2n}\mathbf{x}_{2n} \\ & & \ddots & \\ z_{m1}\mathbf{x}_{m1} & z_{m2}\mathbf{x}_{m2} & \cdots & z_{mn}\mathbf{x}_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \vdots \\ \mathbf{P}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

Points ($4 \times n$)

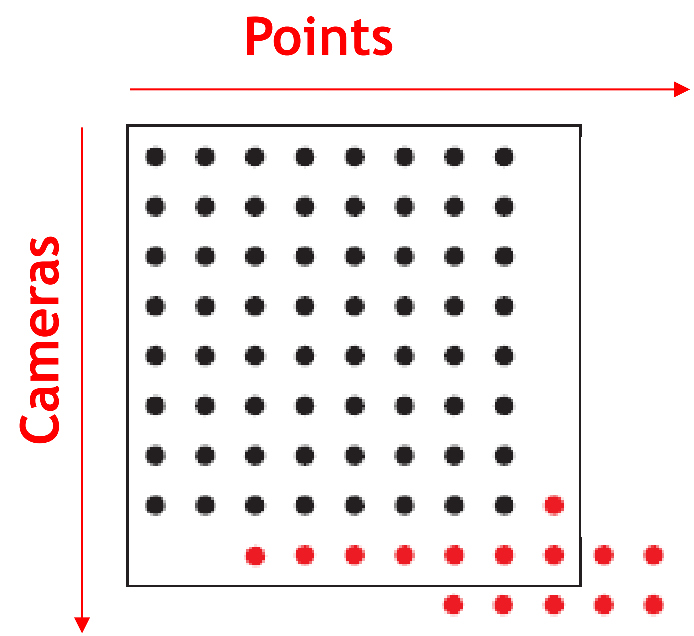
Cameras ($3m \times 4$)

$\mathbf{D} = \mathbf{MS}$ has rank 4

- If we knew the depths *z*, we could factorize D to estimate M and S.
- If we knew M and S, we could solve for *z*.
- Solution: iterative approach (alternate between above two steps).

Recap: Sequential Projective SfM

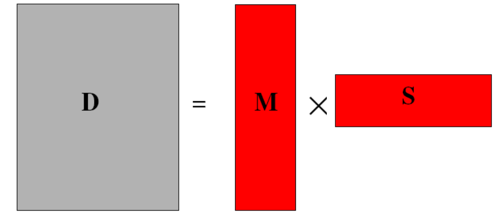
- Initialize motion from two images using fundamental matrix
- Initialize structure
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image - *calibration*
  - Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera - *triangulation*
- Refine structure and motion: *bundle adjustment*



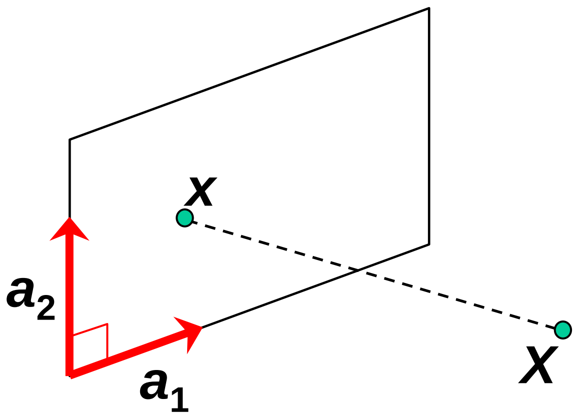
Slide credit: Svetlana Lazebnik

Recap: Estimating the Euclidean Upgrade

- Goal: Estimate ambiguity matrix C
  - Orthographic assumption:



M → MC, S → C^{-1}S



1) Image axes are perpendicular

$$a_1 \cdot a_2 = 0$$

2) Scale is 1

$$|a_1|^2 = |a_2|^2 = 1$$

- This can be converted into a system of 3*m* equations:

$$\begin{cases} \hat{a}_{i1} \cdot \hat{a}_{i2} = 0 \\ |\hat{a}_{i1}| = 1 \\ |\hat{a}_{i2}| = 1 \end{cases} \Leftrightarrow \begin{cases} a_{i1}^T C C^T a_{i2} = 0 \\ a_{i1}^T C C^T a_{i1} = 1, \quad i = 1, \ldots, m \\ a_{i2}^T C C^T a_{i2} = 1 \end{cases}$$

with $L = CC^T$ this translates to

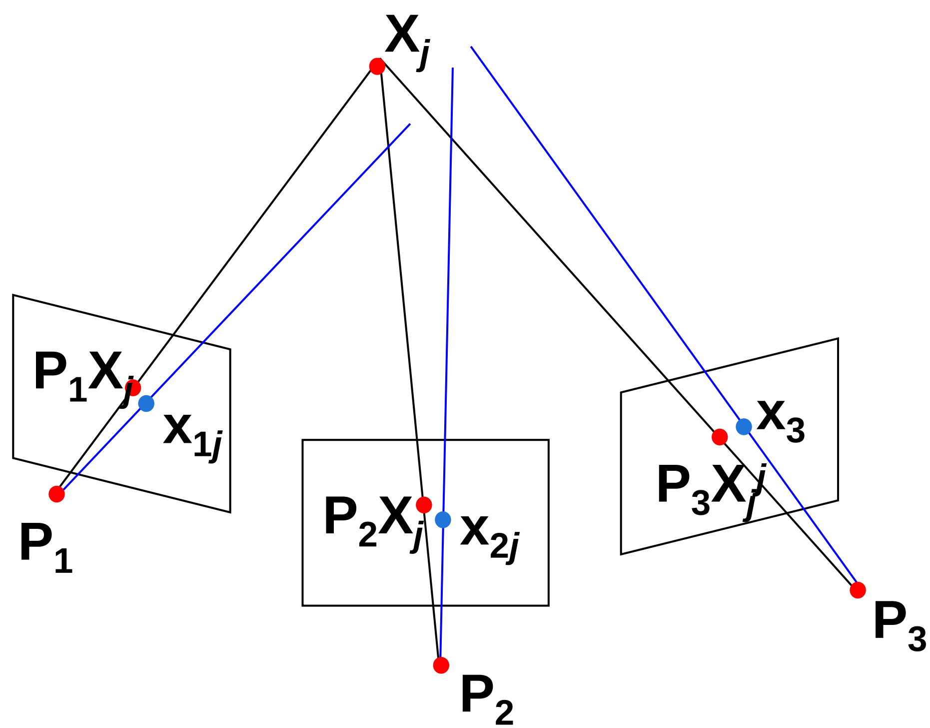
$$A_i L A_i^T = I$$

Slide adapted from S. Lazebnik, M. Hebert

Recap: Bundle Adjustment

- Non-linear method for refining structure and motion
- Minimizing mean-square reprojection error

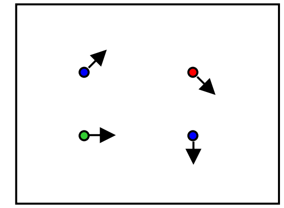
$$E(\mathbf{P},\mathbf{X}) = \sum_{i=1}^{m}\sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i\mathbf{X}_j\right)^2$$



Slide credit: Svetlana Lazebnik

Repetition

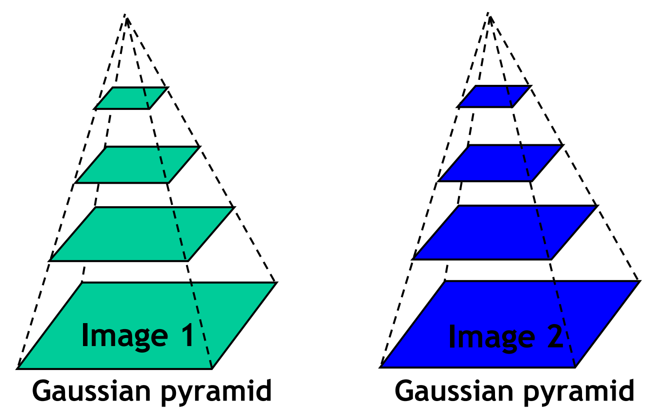
- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking
  - Motion and Optical Flow



Motion field

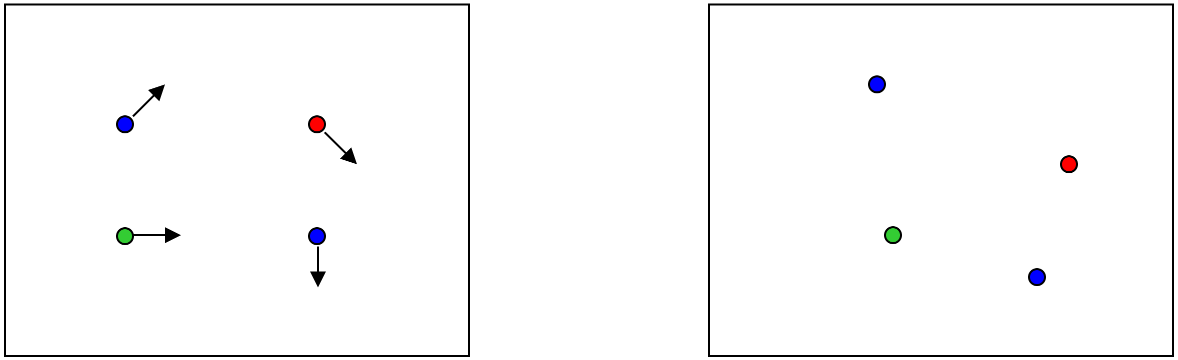
$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

Lucas-Kanade optical flow



Coarse-to-fine estimation

Recap: Estimating Optical Flow



$I(x,y,t-1)$ $I(x,y,t)$

- Given two subsequent frames, estimate the apparent motion field *u*(*x*,*y*) and *v*(*x*,*y*) between them.
- Key assumptions
  - Brightness constancy: projection of the same point looks the same in every frame.
  - Small motion: points do not move very far.
  - Spatial coherence: points move like their neighbors.

Recap: Lucas-Kanade Optical Flow

see Exercise 6.4!

- Use all pixels in a K×K window to get more equations.
- Least squares problem:

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix}$$

$$\underset{25\mathrm{x}2}{A} \quad \underset{2\mathrm{x}1}{d} = \underset{25\mathrm{x}1}{b}$$

- Minimum least squares solution given by solution of

$$\underset{2\mathrm{x}2}{(A^T A)} \; \underset{2\mathrm{x}1}{d} = \underset{2\mathrm{x}1}{A^T b}$$

Recall the Harris detector!

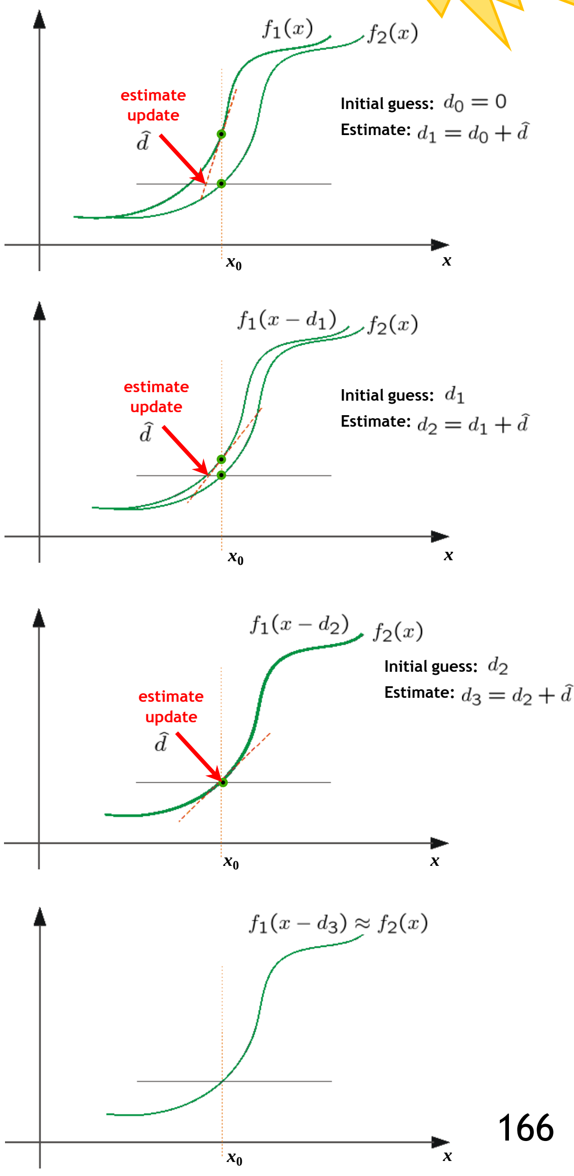
$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

Recap: Iterative Refinement

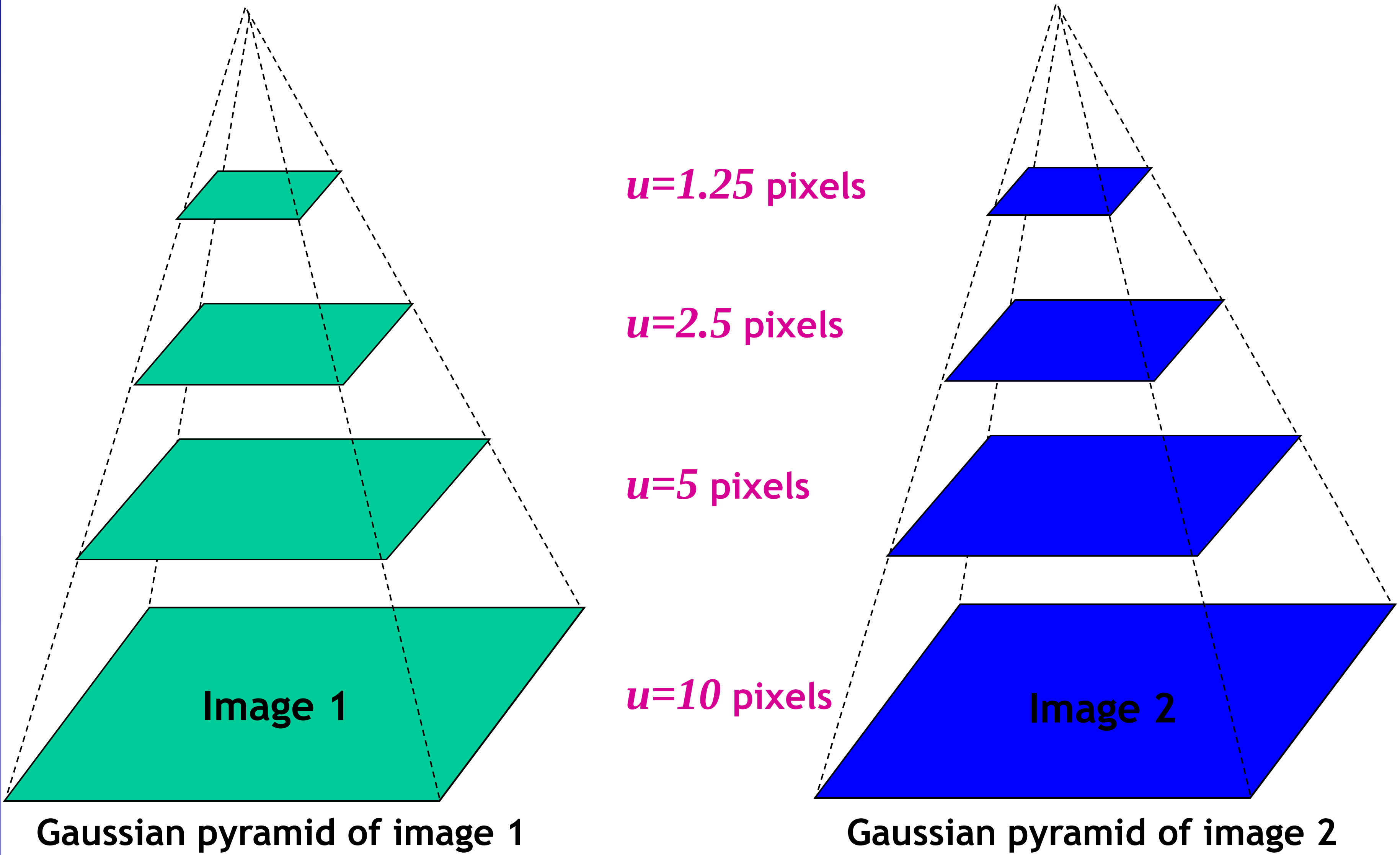
see Exercise 6.4!

- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.

- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.



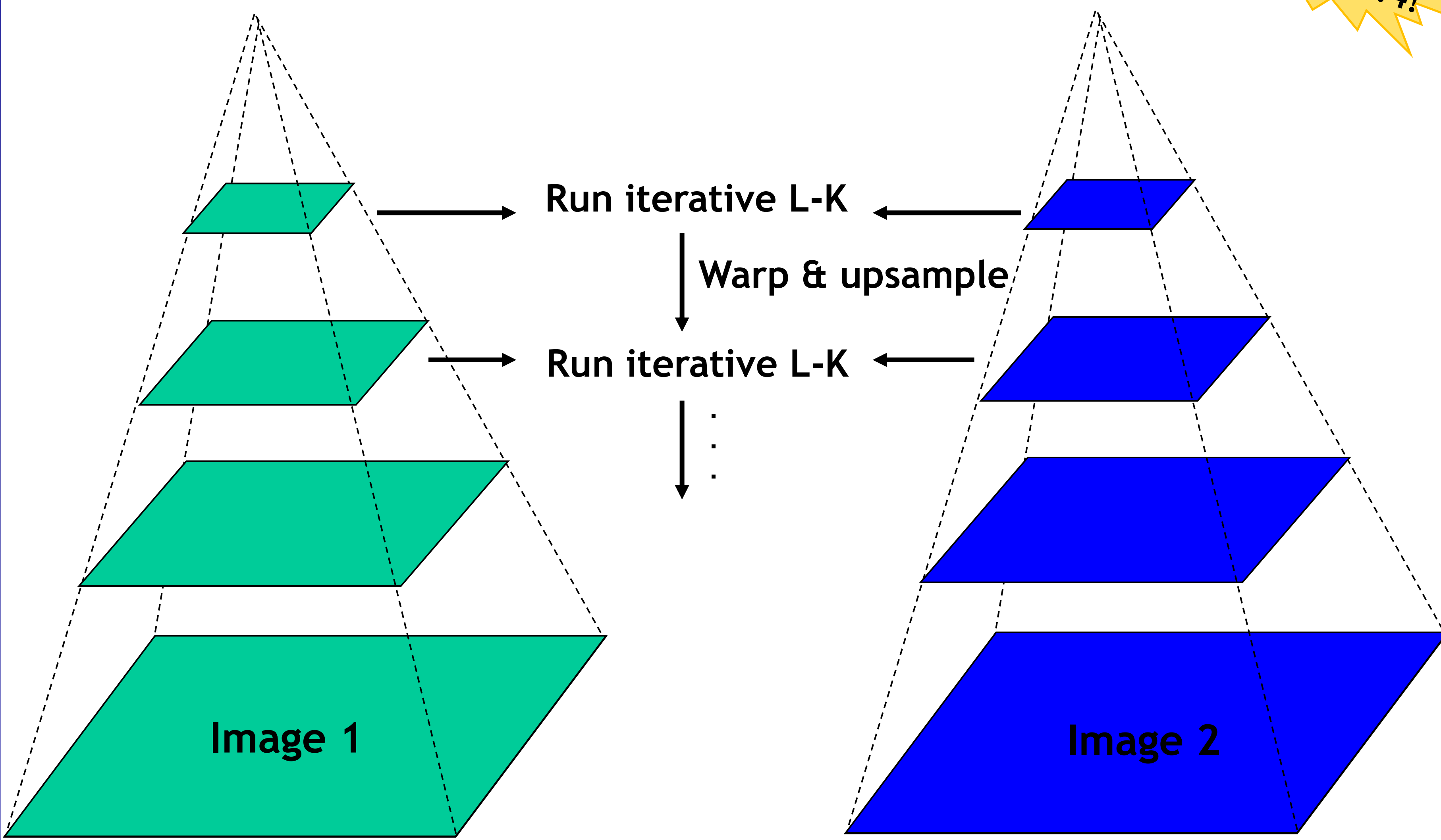
Recap: Coarse-to-fine Estimation



Slide credit: Steve Seitz

Recap: Coarse-to-fine Estimation

see Exercise 6.4!



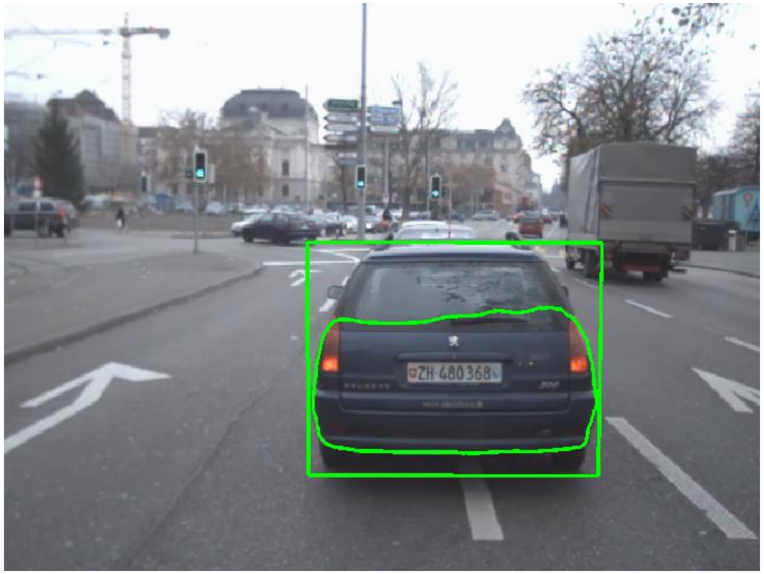
Gaussian pyramid of image 1

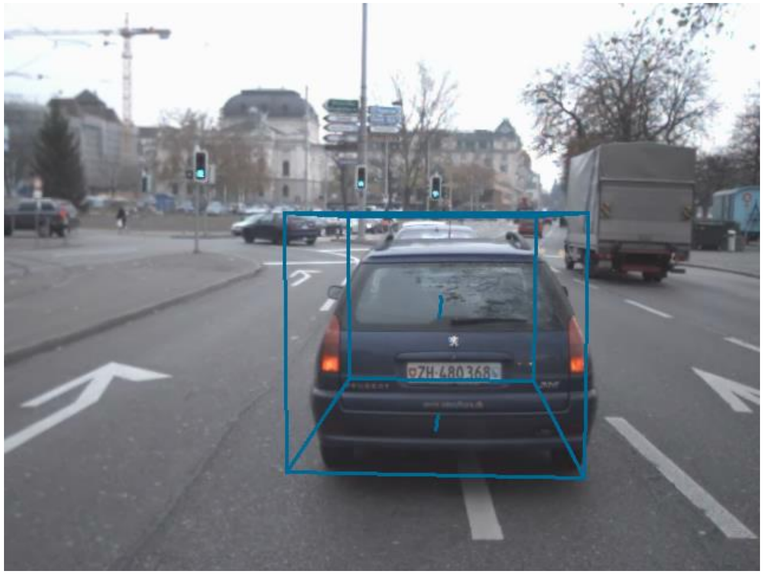
Gaussian pyramid of image 2

Any Questions?

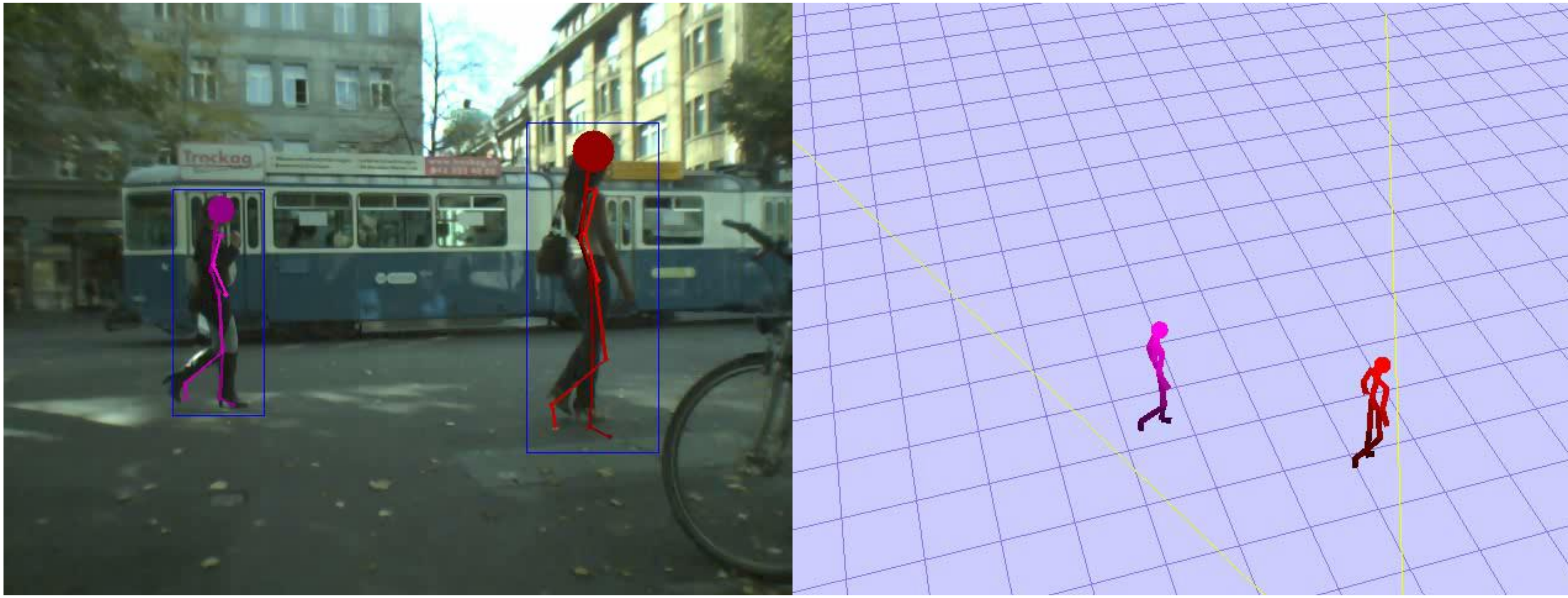
So what can you do with all of this?

Robust Object Detection & Tracking





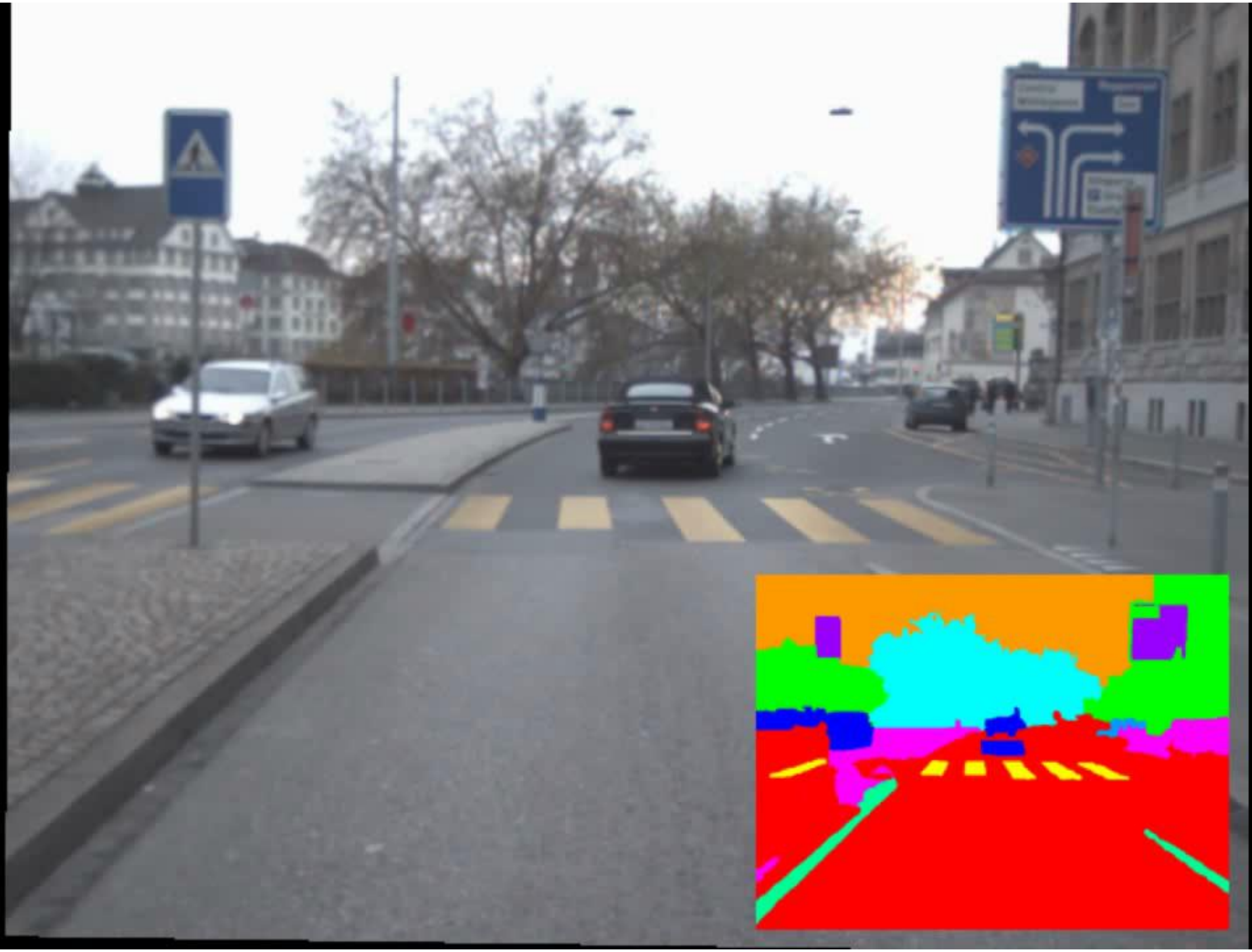
Articulated Multi-Person Tracking



- **Multi-Person tracking**
  - Recover trajectories and solve data association
- **Articulated Tracking**
  - Estimate detailed body pose for each tracked person

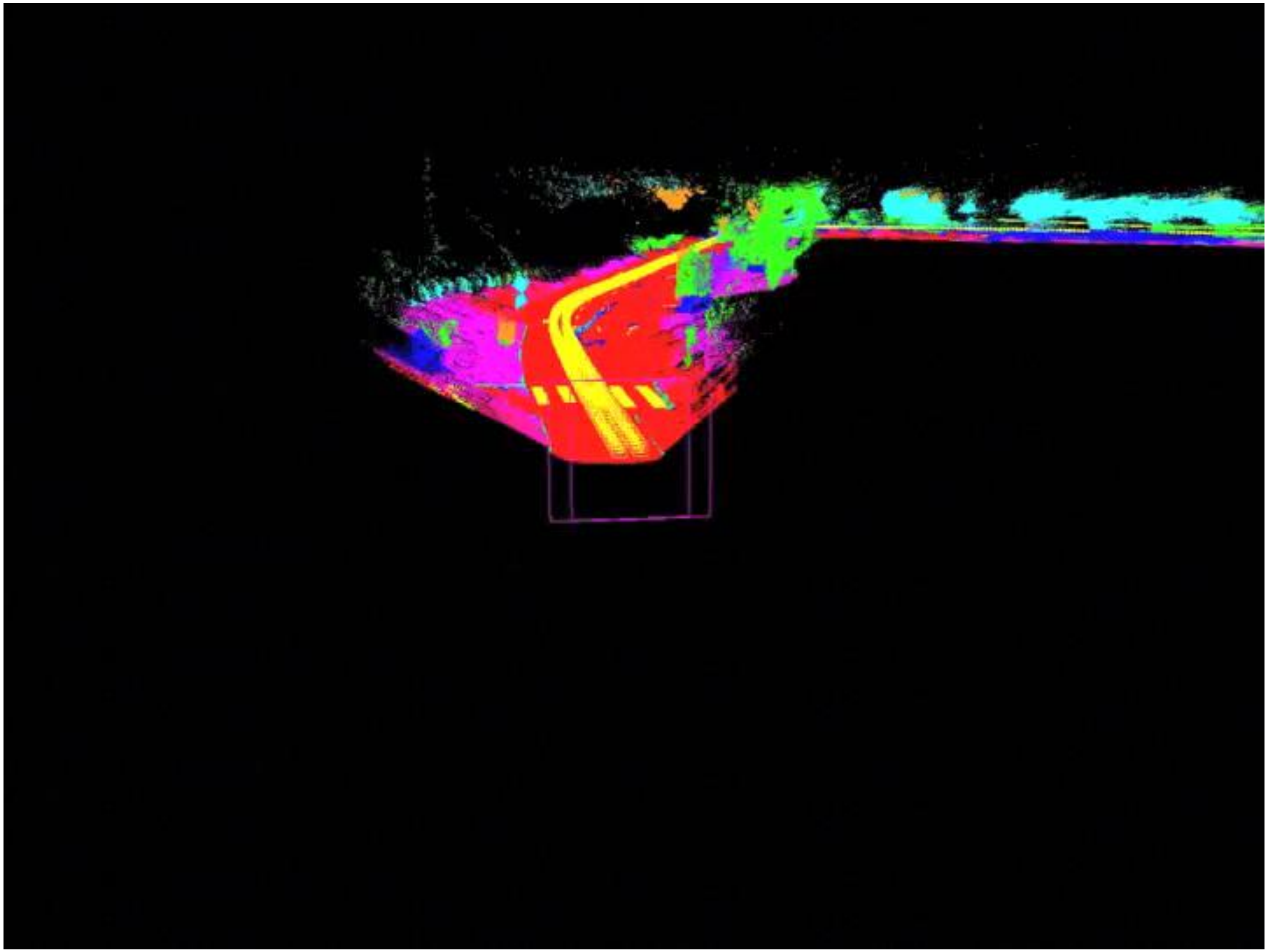
[Gammeter, Ess, Jaeggli, Schindler, Leibe, Van Gool, ECCV'08]

Semantic 2D-3D Scene Segmentation



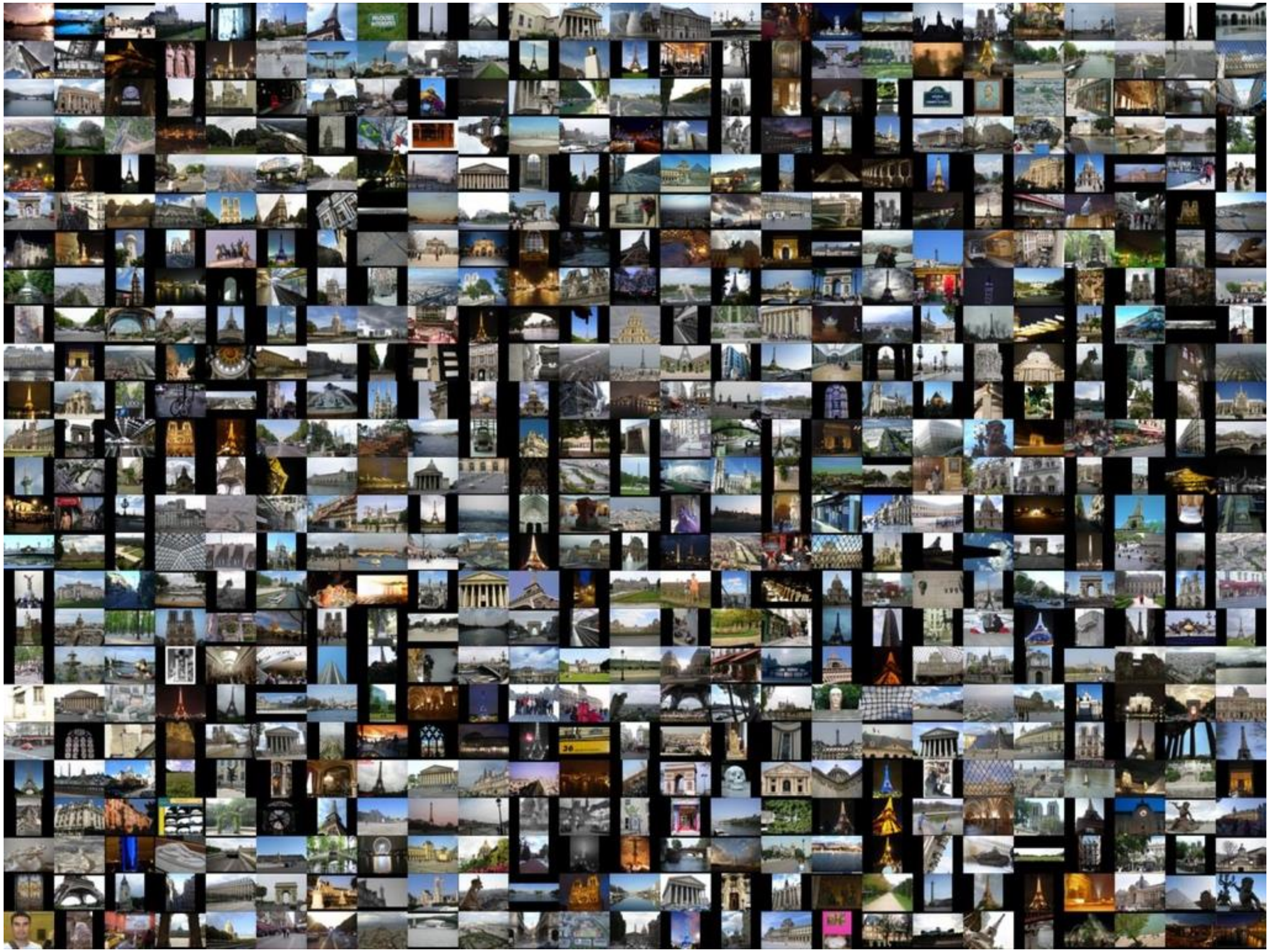
[G. Floros, B. Leibe, CVPR'12]

Integrated 3D Point Cloud Labels

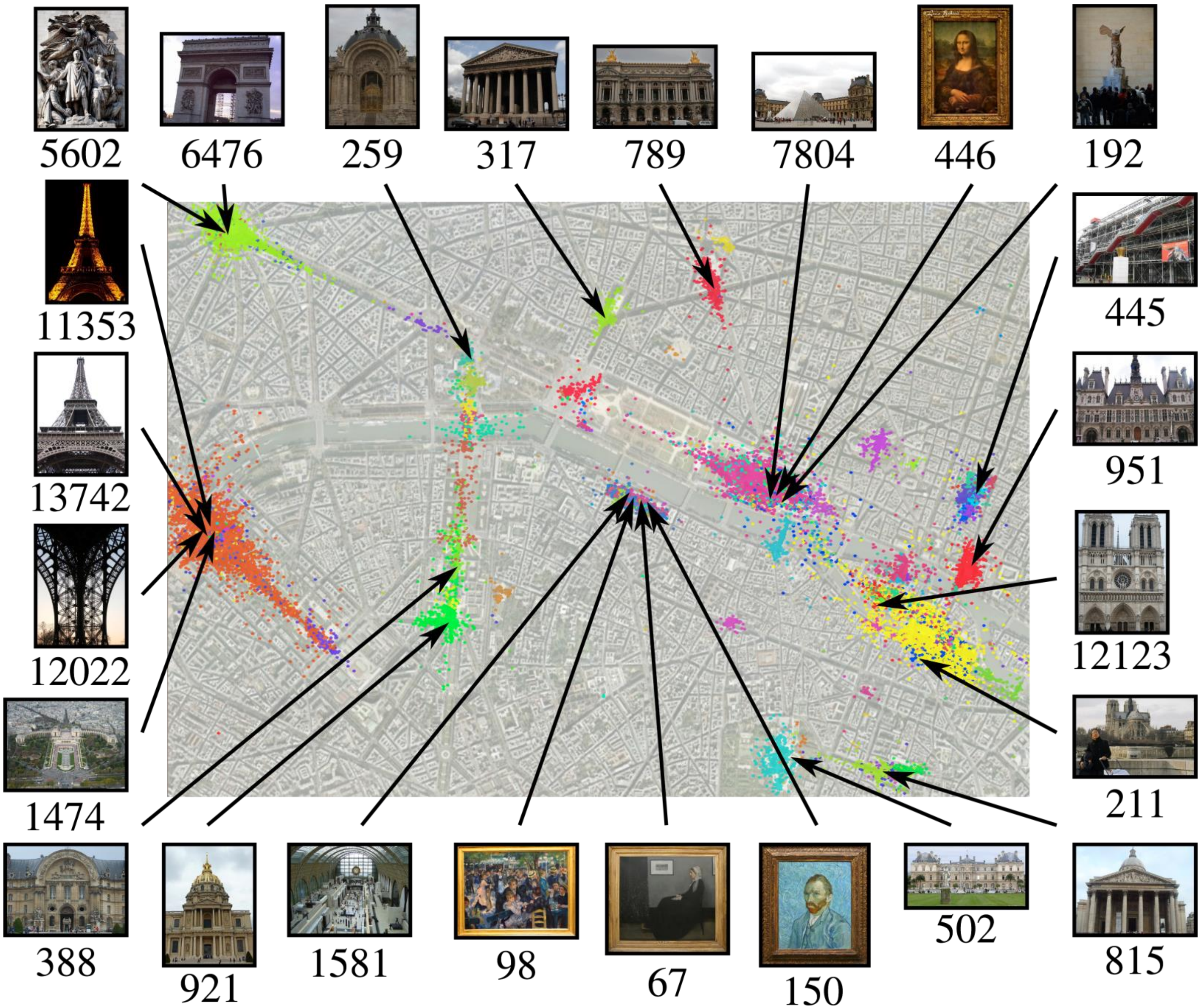


[G. Floros, B. Leibe, CVPR'12]

Mining the World's Images…



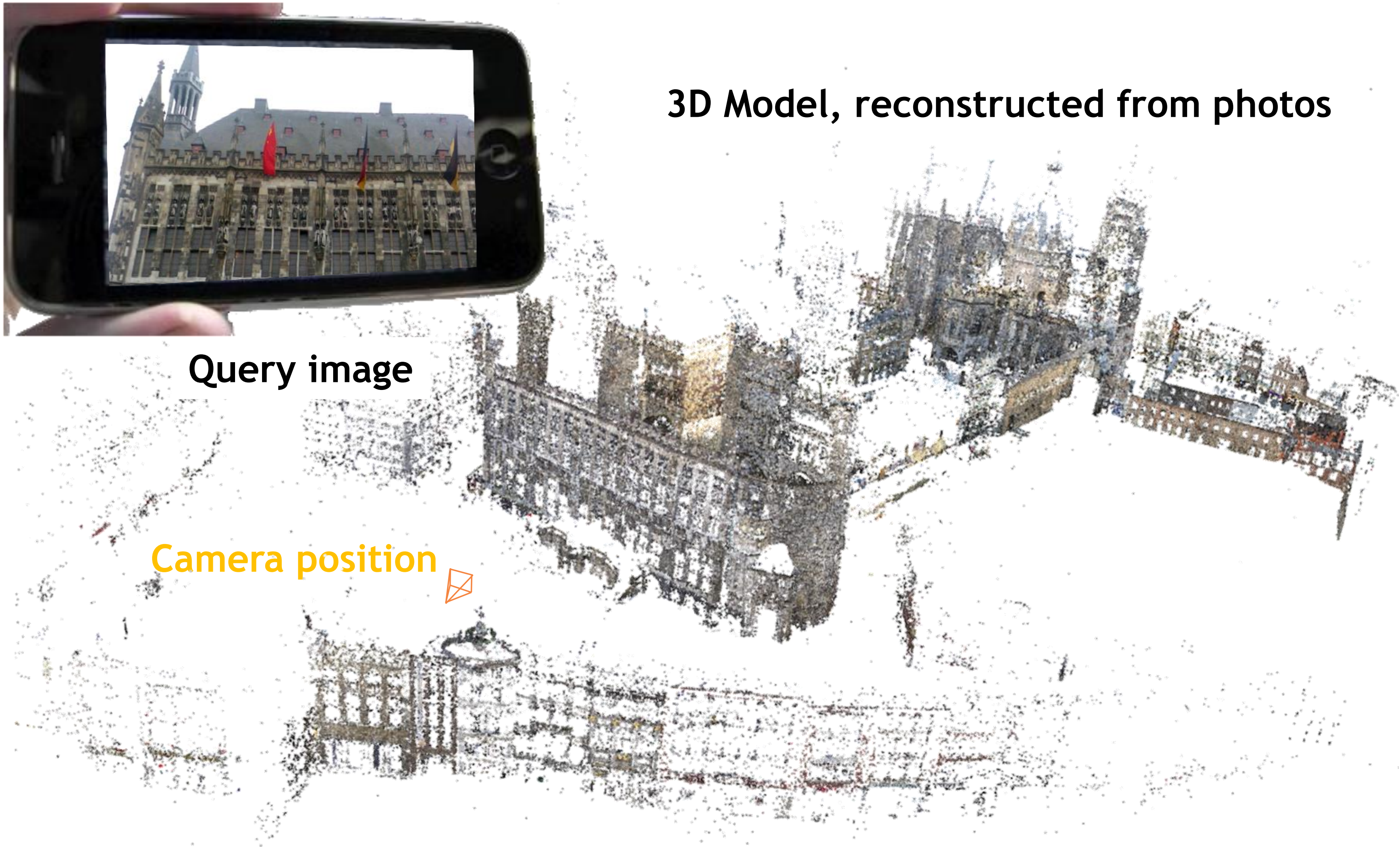
Automatic Landmark Building Discovery



Mobile Visual Search & Mobile AR

- Tourist Guide Scenario
  - Simply point the camera to any object/building of interest.
  - Images are transmitted to a central server for recognition.
  - Object-specific information is sent back to be displayed on the mobile phone.
  - Mobile Augmented Reality fusion of graphics with real video.

Efficient Large-Scale Localization



[T. Sattler, B. Leibe, L. Kobbelt, ICCV'11]

Any More Questions?

Good luck for the exam!